(19) (11) Publication number: **0**

PATENT ABSTRACTS OF JAPAN

(21) Application number: **06232775**

(51) Intl. Cl.: **G06F 3/06** G06F 3/06 G06F 12/16

(22) Application date: **28.09.94**

(30) Priority: **24.12.93 JP 05326823**

(43) Date of application publication: **29.08.95**

(84) Designated contracting states:

(71) Applicant: **HITACHI LTD**
HITACHI COMPUT EN LTD

(72) Inventor: **YAMASHITA YOJI**
TAKAHASHI HIDEO
HATAKEYAMA ATSUS
KATO KANJI
TAKEMURA HIROSHI
URATANI IKUO
KITO AKIRA
MAKI TOSHIYUKI
YAMADA HIDENORI
SHIROTA KOJI
TAKARA AKIKO

(74) Representative:

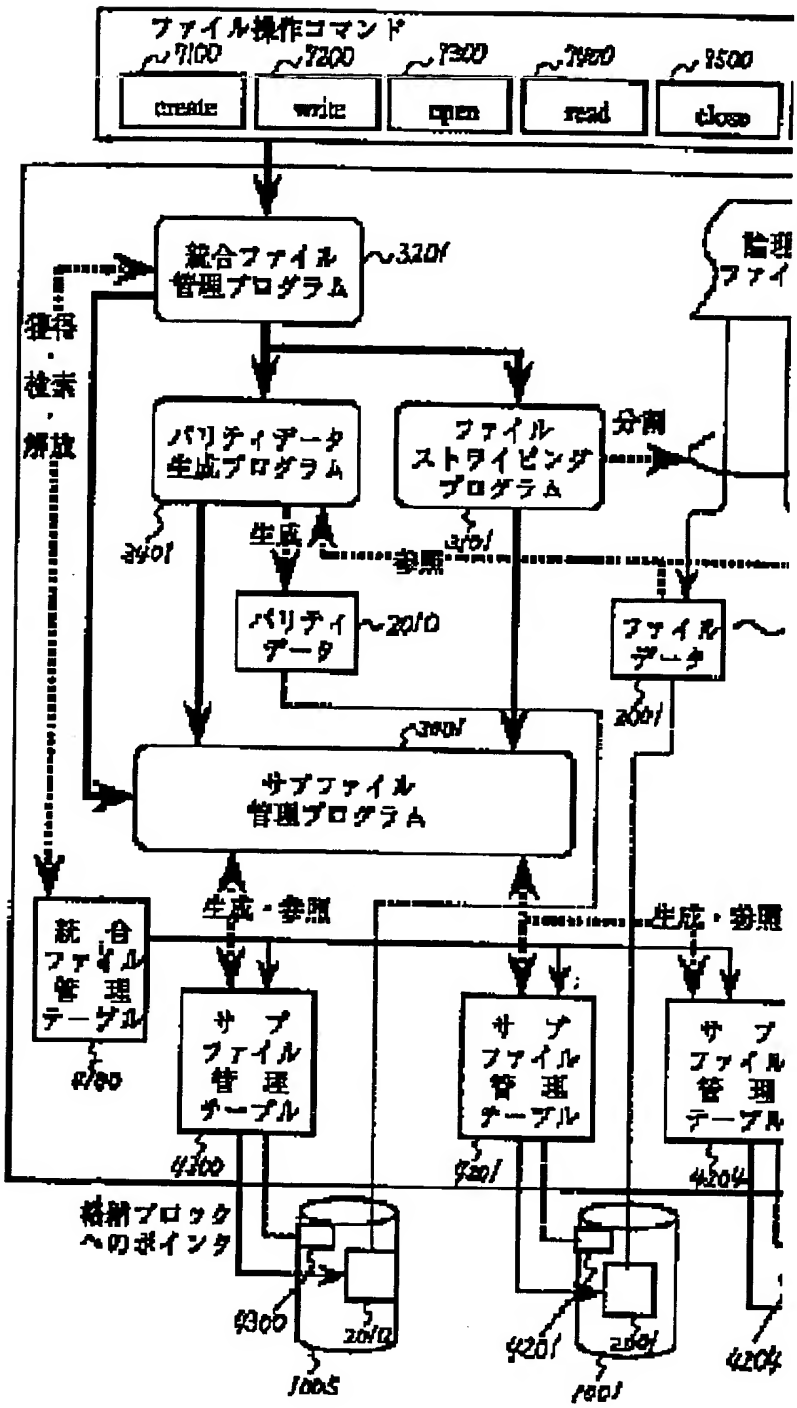
(54) METHOD AND DEVICE FOR FILE MANAGEMENT

(57) Abstract:

PURPOSE: To provide the method and the device for file management of high performance and high reliability.

CONSTITUTION: The file management device is provided with a file striping means 3101, which divides data of a file into plural data, and a subfile management means 3301 which uses sub-file management tables 4201 to 4204 and 4300 to manage divided data of the file with respect to each disk device. The device consists of a parity data

generating means 3401, which operates exclusive OR of data at intervals of the same byte displacement from the head position of each subfile to generate parity data, and an integrated file management means 3201 which uses an integrated file management table 4100 to manage the file.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報(A)　(11)特許出願公開番号

特開平7－230361

(43)公開日　平成７年(1995)８月29日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　3/06	５４０			
	３０５　Ｃ			
12/00	５３１　Ｄ	7608－5B		
12/16	３２０　Ｌ	7608－5B		

審査請求　未請求　請求項の数30　ＯＬ　（全 27 頁）

(21)出願番号　特願平6－232775

(22)出願日　平成６年(1994)９月28日

(31)優先権主張番号　特願平5－326823

(32)優先日　平５(1993)12月24日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(71)出願人　000233011
日立コンピュータエンジニアリング株式会社
神奈川県秦野市堀山下１番地

(72)発明者　山下　洋史
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内

(74)代理人　弁理士　小川　勝男

最終頁に続く

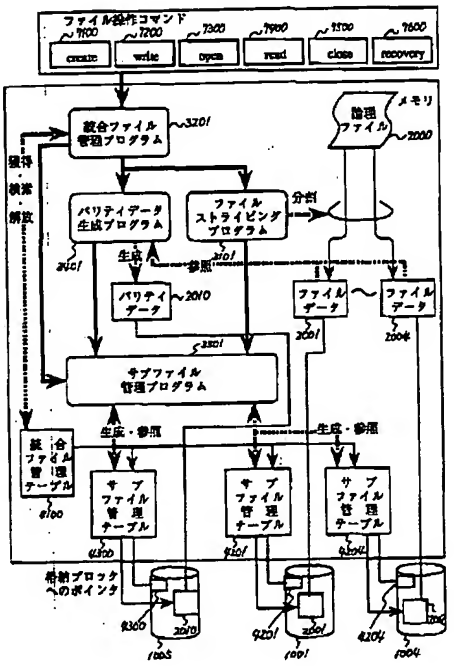
(54)【発明の名称】　ファイル管理方法及び装置

(57)【要約】

【目的】　高性能、高信頼なファイル管理方法及びファイル管理装置を提供する。

【構成】　本発明のファイル管理装置は、ファイルのデータを複数のデータに分割するファイルストライピング手段３１０１、分割されたファイルのデータをディスク装置ごとにサブファイル管理テーブル４２０１～４２０４、４２３０を用いて管理するサブファイル管理手段３３０１、各サブファイルの先頭位置から同一のバイト変位のデータごとに排他的論理和を取ってパリティデータを生成するパリティデータ生成手段３４０１、統合ファイル管理テーブル４１００を用いてファイルを管理する統合ファイル管理手段３２０１から構成される。

図1



1

【特許請求の範囲】

【請求項１】ファイルデータを格納する複数の二次記憶装置を備えたファイル管理装置において、

第一のファイルのデータを複数の第二のデータに分割するファイルストライピング手段と、

前記第二のデータのそれぞれの先頭位置から同一バイト変位のバイトごとのデータの排他的論理和をパリティデータとして生成するパリティデータ生成手段と、　前記第二のデータと前記パリティデータを格納する前記複数の二次記憶装置内のブロックのブロック番号を前記二次記憶装置の個別ごとに管理するサブファイル管理テーブルを用いて前記第二のデータを管理するサブファイル管理手段と、

前記サブファイル管理テーブルの全てと前記パリティデータへのポインタとを管理する統合ファイル管理テーブルを用いて前記第一のファイルを管理する統合ファイル管理手段とを設けたことを特徴とするファイル管理装置。

【請求項２】請求項１記載のファイル管理装置は、ファイル利用手段を有し、

前記ファイル利用手段は、ファイルを新規に作成するファイル新規作成命令、ファイルデータ書き込み命令、ファイル操作を行うために統合ファイル管理テーブルを獲得するファイルオープン命令、ファイルのデータを読み出すファイル読み出し命令、前記統合ファイル管理テーブルを解放するファイルクローズ命令、および故障した二次記憶装置のデータにアクセスするためのファイル回復命令を実行することを特徴とするファイル管理装置。

【請求項３】請求項１記載のファイル管理装置において、

前記統合ファイル管理手段は、未使用の統合ファイル管理テーブルを割り当てる統合ファイル管理テーブル割り当て手段、前記統合ファイル管理テーブルを検索し獲得する統合ファイル管理テーブル検索手段、前記統合ファイル管理テーブル割り当て手段と前記統合ファイル管理テーブル検索手段によって獲得された前記統合ファイル管理テーブルを解放する統合ファイル管理テーブル解放手段を設けたことを特徴とするファイル管理装置。

【請求項４】請求項１記載のファイル管理装置において、

前記統合ファイル管理テーブルは、ユーザがファイルを一意に指定するための記述子を格納するエントリ、分割されたファイルデータを管理するサブファイル管理テーブルへのポインタを格納するエントリ、およびパリティデータを管理するサブファイル管理テーブルへのポインタを格納するエントリを設けたことを特徴とするファイル管理装置。

【請求項５】請求項３記載のファイル管理装置において、

前記統合ファイル管理テーブル割り当て手段は、未使用

2

の統合ファイル管理テーブルを検索する手段、未使用な統合ファイル管理テーブルが無い場合には未使用の統合ファイル管理テーブルができるまで待機する手段、および未使用の統合ファイル管理テーブルを獲得する手段を設けたことを特徴とするファイル管理装置。

【請求項６】請求項３記載のファイル管理装置において、

前記統合ファイル管理テーブル検索手段は、ファイル名称をキーに統合ファイル管理テーブルを検索する手段を設けたことを特徴とするファイル管理装置。

【請求項７】請求項１記載のファイル管理装置において、

前記サブファイル管理手段は、未使用のサブファイル管理テーブルを割り当てるサブファイル管理テーブル割り当て手段、サブファイル管理テーブルを検索し獲得するサブファイル管理テーブル検索手段、前記サブファイル管理テーブル割り当て手段とサブファイル管理テーブル検索手段で獲得されたサブファイル管理テーブルを解放するサブファイル管理テーブル解放手段、および分割されたファイルデータやパリティデータを格納する二次記憶装置内の位置を決定するデータ格納位置決定手段を設けたことを特徴とするファイル管理装置。

【請求項８】請求項１記載のファイル管理装置において、

前記サブファイル管理テーブルは、ユーザがファイルを一意に指定するための記述子を格納するエントリ、同一ディスク装置内のサブファイル管理テーブルの識別子を格納するエントリ、二次記憶装置番号を格納するエントリと，分割されたファイルデータやパリティデータが二次記憶装置内のどの位置にかくのうされたかを示すエントリ、パリティデータを格納する二次記憶装置を示すフラグ、および前記サブファイル管理テーブルが使用されているかいないかを示すフラグを設けたことを特徴とするファイル管理装置。

【請求項９】請求項７記載のファイル管理装置において、

前記サブファイル管理テーブル割り当て手段は、未使用のサブファイル管理テーブルを検索する手段、未使用なサブファイル管理テーブルが無い場合には待機する手段、および未使用のサブファイル管理テーブルを獲得する手段を設けたことを特徴とするファイル管理装置。

【請求項１０】請求項７記載のファイル管理装置は、さらに、

ファイル名称をキーにサブファイル管理テーブルを検索する手段を設けたことを特徴とするファイル管理装置。

【請求項１１】請求項１記載のファイル管理装置は、さらに、

前記二次記憶装置上のデータをメモリ上にキャッシングするためのバッファキャッシュ、前記バッファキャッシュのうちパリティデータ生成に関係のあるパリティグル

3

ープを管理するパリティグループ管理テーブル、および前記バッファキャッシュとパリティグループ管理テーブルを管理するバッファキャッシュ管理手段を設けたことを特徴とするファイル管理装置。

【請求項１２】請求項１１記載のファイル管理装置において、

前記バッファキャッシュは、前記パリティグループ管理テーブルへのポインタを格納するエントリを設けたことを特徴とするファイル管理装置。

【請求項１３】請求項１１記載のファイル管理装置において、

前記パリティグループ管理テーブルは、前記統合ファイル管理テーブルへのポインタを格納するエントリ、およびバッファキャッシュへのポインタを格納するエントリを設けたことを特徴とするファイル管理装置。

【請求項１４】請求項１１記載のファイル管理装置において、

前記バッファキャッシュ管理手段は、ファイルの論理ブロック番号からパリティグループ番号を計算する手段、パリティグループ管理テーブルを検索する手段、パリティグループ管理テーブルを獲得する手段、バッファキャッシュを検索する手段、バッファキャッシュを獲得する手段、パリティ生成を行うバッファキャッシュを選択する手段、および前記選択されたバッファキャッシュのパリティ生成を行うために前記パリティ生成手段を起動する手段を設けたことを特徴とするファイル管理装置。

【請求項１５】請求項８記載のファイル管理装置において、

前記サブファイル管理テーブルに隣接する二次記憶装置のサブファイル管理テーブルのサブファイル管理テーブル番号を格納するエントリを前記サブファイル管理テーブルに設けたことを特徴とするファイル管理装置。

【請求項１６】複数の二次記憶装置にファイルデータを格納するファイル管理方法において、

ａ）第一のファイルのデータを複数の第二のデータに分割し、

ｂ）前記第二のデータのそれぞれの先頭位置から同一バイト変位のバイトごとのデータの排他的論理和をパリティデータとして生成し、

ｃ）前記第二のデータと前記パリティデータを格納する前記複数の二次記憶装置内のブロックのブロック番号を前記二次記憶装置の個別ごとに管理するサブファイル管理テーブルを用いて前記第二のデータを管理し、さらに、

ｄ）前記サブファイル管理テーブルの全てと前記パリティデータへのポインタを管理する統合ファイル管理テーブルを用いて前記第一のファイルを管理することを特徴とするファイル管理方法。

【請求項１７】請求項１６記載のファイル管理方法において、

4

ユーザのファイル操作命令として、ファイルを新規に作成するファイル新規作成命令、ファイルデータ書き込み命令、ファイル操作を行うために統合ファイル管理テーブルを獲得するファイルオープン命令、ファイルのデータを読み出すファイル読み出し命令、前記統合ファイル管理テーブルを解放するファイルクローズ命令、および故障した前記二次記憶装置のデータにアクセスするためのファイル回復命令を受け付けることを特徴とするファイル管理方法。

【請求項１８】請求項１６記載のファイル管理方法において、前記ステップｄ）は、

ｄ１）未使用の統合ファイル管理テーブルを割り当てるステップ、

ｄ２）前記統合ファイル管理テーブルを検索し獲得する統合ファイル検索ステップ、および

ｄ３）前記獲得された統合ファイル管理テーブルを解放するステップを有することを特徴とするファイル管理方法。

【請求項１９】請求項１８記載のファイル管理方法において、前記ステップｄ１）は、

ｄ１１）未使用の統合ファイル管理テーブルを検索し、

ｄ１２）未使用な統合ファイル管理テーブルが無い場合には未使用の統合ファイル管理テーブルができるまで待機し、さらに、

ｄ１３）未使用の統合ファイル管理テーブルを獲得することを特徴とするファイル管理方法。

【請求項２０】請求項１９記載のファイル管理方法において、前記ステップｄ１１）は、

ファイル名称をキーに統合ファイル管理テーブルを検索することを特徴とするファイル管理方法。

【請求項２１】請求項１６記載のファイル管理方法において、前記ステップｃ）は、

ｃ１）未使用のサブファイル管理テーブルを割り当てるステップ、

ｃ２）サブファイル管理テーブルを検索し獲得するステップ、

ｃ３）前記獲得されたサブファイル管理テーブルを解放するステップ、および、　ｃ４）分割されたファイルデータやパリティデータを格納する二次記憶装置内の位置を決定することを特徴とするファイル管理方法。

【請求項２２】請求項２１記載のファイル管理方法において、前記ステップｃ１）は、

ｃ１１）未使用のサブファイル管理テーブルを検索し、

ｃ１２）未使用なサブファイル管理テーブルが無い場合には待機し、

ｃ１３）未使用のサブファイル管理テーブルを獲得することを特徴とするファイル管理方法。

【請求項２３】請求項２１記載のファイル管理方法は、さらに、

ファイル名称をキーにサブファイル管理テーブルを検索

5

することを特徴とするファイル管理方法。

【請求項２４】請求項１７記載のファイル管理方法は、ユーザからファイル作成命令が発効されると、パリティデータを格納する二次記憶装置を決定することを特徴とするファイル管理方法。

【請求項２５】請求項２４記載のファイル管理方法は、さらに、

前記二次記憶装置のそれぞれの使用量を調べ、

前記使用量からパリティデータを格納する二次記憶装置を決定することを特徴とするファイル管理方法。

【請求項２６】請求項２４記載のファイル管理方法は、さらに、

前記二次記憶装置のそれぞれのアクセス頻度を調べ、

前記アクセス頻度からパリティデータを格納する二次記憶装置を決定することを特徴とするファイル管理方法。

【請求項２７】請求項１６記載のファイル管理方法は、さらに、

パリティデータの書き込みを開始する二次記憶装置を決定し、

１ブロックごとに決定した二次記憶装置から順番にパリティデータの残りの二次記憶装置に巡回させることを特徴とするファイル管理方法。

【請求項２８】請求項１６記載のファイル管理方法は、さらに、

二次記憶装置上のデータをメモリ上にキャッシングするためのバッファキャッシュに格納し、

前記バッファキャッシュのうちパリティデータ生成に関係のあるパリティグループを管理するための情報をパリティグループ管理テーブルに格納し、

前記バッファキャッシュとパリティグループ管理テーブルを管理することを特徴とするファイル管理方法。

【請求項２９】請求項２８記載のファイル管理方法において、前記バッファキャッシュの管理を行うステップは、

ファイルの論理ブロック番号からパリティグループ番号を計算し、

パリティグループ管理テーブルを検索して、前記パリティグループ管理テーブルを獲得し、

前記バッファキャッシュを検索して、バッファキャッシュを獲得し、

パリティ生成を行うバッファキャッシュを選択し、

前記選択されたバッファキャッシュのパリティ生成を行うことを特徴とするファイル管理方法。

【請求項３０】請求項１６記載のファイル管理方法において、

前記サブファイル管理テーブルに隣接する二次記憶装置のサブファイル管理テーブルのサブファイル管理テーブル番号を格納するエントリに前記管理テーブル番号を格納し、

前記管理テーブル番号に基づいて他のサブファイル管理

6

テーブルを獲得することを特徴とするファイル管理方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明はワークステーション等の情報処理装置で用いる複数の二次記憶装置にファイルのデータを格納するためのファイル管理方法およびそれを用いたファイル管理装置に係り、特に、二次記憶装置が故障した場合でもファイルのデータを読み出すことができる高信頼なファイル管理方法およびファイル管理装置に関する。

【０００２】

【従来の技術】本発明に関する従来技術として，D.A.Patterson, et al. "Introduction to Redundant Arrays of Inexpensive Disks(RAID)", spring COMPCON'89, pp.112-117, Feb.1989がある。Pattersonの論文は、RAIDのデータ配置に関する技術を開示した論文である。RAIDは、物理的には複数のディスク装置からなるディスクシステムをあたかも一台のディスク装置として扱うことにより、高性能化／高信頼化を図る機構である。

【０００３】Pattersonの論文では、いくつかのデータ配置方法が提案されているが、代表的なデータ配置方法として、処理装置とのリード／ライト単位であるブロックをそのままの形で配置する方法がある。この方法の特長は、RAIDを構成するそれぞれのディスク装置ごとにリード／ライト処理が実行可能な点である。これにより、RAID内で実行できるリード／ライト処理の多重度を向上させることができ、性能向上が実現できる。また、Pattersonの論文で提案されている他の配置方法として、ブロックを複数のディスク装置に分割して配置するという方法も提案されている。この方法の特長は、長大データをシーケンシャルにアクセスする場合に複数のディスク装置を並列に動作させることができる点である。これにより、長大データの高速アクセスが実現できる。

【０００４】一方、RAIDの高信頼化は、パリティデータと呼ばれる冗長データをディスク装置に格納することによって実現される。RAIDを構成する複数のディスク装置のそれぞれの同一物理ブロック番号のブロックから、ブロックに相当するデータ量のパリティデータが作成され、作成されたパリティデータは、一つのブロックとして別のディスク装置の同一物理ブロック番号のブロックに格納される。これをパリティブロックと呼ぶ。また、パリティブロックとそれを生成したブロックの集合をパリティグループと呼ぶ。さらに、パリティブロック以外のブロックをデータブロックと呼ぶ。

【０００５】パリティブロックには、パリティブロックを生成したデータブロックのうちのどれか一つのブロックに障害が発生しても、パリティグループの他のブロックの内容から復元可能にするための内容が格納されている。通常は、パリティグループに含まれる各データブロ

7

ックの排他的論理和がパリティブロックとして格納される。

【０００６】上記のデータ配置方法の場合、書き込み性能が単体のディスク装置の性能に比較して劣化するという問題が発生する。すなわち、書き込み処理が発生したデータブロックの更新値以外にパリティブロックの更新値も書き込む必要があるため、書き込み性能が劣化する。しかも、パリティブロックの更新値を決定するためには、書き込み処理が発生したデータブロックの更新値以外に、以下に示す二つの前処理のいずれかを実行する必要がある。

【０００７】（1）書き込み処理対象ブロックの更新前の値と、パリティブロックの更新前の値のディスク装置からの読み出し、（2）パリティグループの他のブロックの値を参照する。

【０００８】（1）と（2）の処理のうち、ディスク装置からの読み出しブロック数の少ない方が実行される。パリティグループに含まれる大半のブロックに対して書き込み処理を行う場合には処理（2）が実行されるが、１ブロックだけの書き込み処理が行われる場合には、読み出し回数が少ない処理（1）が実行される。

【０００９】後者の場合、処理（1）では２ブロックのブロックI/Oが発生し、書き込み処理を行うデータブロックとパリティブロックの更新に２ブロックのブロックI/Oが発生し、計４ブロックのI/Oが発生するため、単体ディスク装置の１／４まで性能が低下する。このように、書き込み時に性能が低下することをライトペナルティと呼ぶ。

【００１０】RAIDにおいてライトペナルティを解消して書き込み処理を高速化する技術として、特開平4-245352(記憶装置制御方法および記憶装置サブシステム)には、ディスクキャッシュを有するRAID装置において、非同期に書き込み要求を処理することにより、書き込み処理の高速化を図る技術が示されている。

【００１１】具体的には、処理装置から書き込み要求を受け取ると、パリティブロックの更新値を作成するのに必要な情報がキャッシュに存在するか否かが識別され、書き込みデータが受け取られると、処理装置に書き込み要求の完了が報告される。処理装置からの書き込み要求に対して、パリティレコードの更新値を作成するために非同期に用意できる情報の中でキャッシュにない情報を読み出す処理とパリティブロックの更新値の書き込み処理とが非同期に行われる。

【００１２】M.Kitsuregawa and K.mogi,"A Raid5 Storage Management Scheme with Robustness for the Peak Access Traffic, International Symposium on Next Database Systems and Their Applications, Fukuoka, Japan September, 1993, p.99-106 "では、従来のように、同一の物理ブロック番号どうしのブロックで固定的にパリティグループを編成せずに、最近書き込まれたブロックどうしで一つのパリティグループを編成して、動的にパリティグループを編成することにより、処理（1）の手間を低減させて高速化を図ることが示されている。

8

【００１３】

【発明が解決しようとする課題】上記のように、RAIDでは、書き込み性能が低下することが問題である。特に、１ブロックだけの書き込み処理が発生する場合には、単体ディスク装置の１／４にまで性能が低下する。

【００１４】さらに、１ブロックだけのデータ書き込みは、ランダム書き込みだけでなく、ファイルが連続的にディスク装置に格納できずに断片化される場合のシーケンシャル書き込みでも発生する。ファイルの断片化は、ファイルの作成や削除を繰り返すことにより頻繁に生じる。そのため、ライトペナルティの低減を図るためには、断片化されたファイルのシーケンシャル書き込みについても考慮する必要がある。

【００１５】特開平4-245352では、RAID制御装置にキャッシュを設け、さらに書き込み要求を非同期化することにより書き込み性能の向上を図っているが、この技術は１ブロックの書き込み要求が発生した後、同一パリティグループの他のブロックへの書き込み要求が後から発生することを前提とした高速化の手法である。すなわち、ファイルが連続的にディスク装置に格納できずに断片化されてしまった場合には、同一パリティグループ内の他のブロックへの書き込み要求が後から発生することは稀であり（同一パリティグループ内の他のブロックに別のファイルのデータが格納されてしまうため）、このような場合には、上記の技術の効果が小さい。

【００１６】また、信学技法CPSY93-32では、異なる物理ブロック番号のブロックをパリティグループに編成しているため、パリティグループがどのブロック番号のブロックで構成されるかを管理する必要があるので処理が複雑になり、さらに定期的にガーベージコレクティングを行う必要があるため、性能が不安定になるという問題がある。

【００１７】本発明で解決すべき課題は、１ブロックだけのデータ書き込み、特に断片化されてしまったファイルのシーケンシャル書き込みで発生する１ブロックだけのデータ書き込みの場合に書き込み処理を効率化することである。すなわち、高速に書き込み処理が行えるRAIDを実現することが本発明の目的である。

【００１８】

【課題を解決するための手段】上記の課題を解決し、断片化されたファイルのシーケンシャル書き込み時に、効率的に書き込み処理を効率的に行うため、本発明は以下の４つのプログラムで構成される。

【００１９】(1)ファイルストライピングプログラム…ファイルデータを格納する複数の二次記憶装置を備えたファイル管理装置において、ファイルのデータを論理ブ

9

ロックごとにそれぞれ複数のデータに分割するプログラム

(2)パリティデータ生成プログラム…ファイルストライピングプログラムにより分割されたデータどうしの先頭バイトから同一バイト変位ごとのデータの排他的論理和をパリティデータとして生成するプログラム

(3)サブファイル管理プログラム…ファイルストライピングプログラムにより分割されたデータとパリティデータ生成プログラムにより生成されたパリティデータとを格納する複数の二次記憶装置内のブロックのブロック番号を、二次記憶装置ごとに管理するサブファイル管理テーブルを設け、このサブファイル管理テーブルを用いてデータを管理するプログラム

(4)統合ファイル管理プログラム…分割されたデータとパリティデータとを管理するサブファイル管理テーブルのポインタを格納する統合ファイル管理テーブルを設け、この統合ファイル管理テーブルを用いてファルを管理するプログラム

【００２０】

【作用】本発明によれば、以下のようにして目的が達成される。本発明では、ファイルストライピングプログラムが、論理ブロック単位でファイルのデータを分割する。さらに、パリティデータ生成プログラムが、分割されたファイルデータの先頭バイトから同一バイト変位ごとの排他的論理和を計算し、パリティデータを生成する。そして、サブファイル管理プログラムが、サブファイル管理テーブルを用いてこれら分割されたファイルのデータとパリティデータとが実際に格納される二次記憶装置上の物理ブロック位置を管理し、統合ファイル管理プログラムが複数のサブファイル管理テーブルを一括して管理する。その結果、ファイルの論理ブロック単位でパリティグループを編成できる。

【００２１】すなわち、物理的にはディスク装置上の連続した領域に格納できずに断片化されてしまったファイルに対してシーケンシャルに書き込みを行う場合に、ファイルの論理ブロック単位でパリティグループを編成しているために、従来、物理ブロックでパリティグループを編成していた場合よりも書き込み処理が効率化される。

【００２２】以下の処理が、パリティ生成の前処理として実行される。

（１）書き込み処理対象ブロックの更新前の値と、パリティブロックの更新前の値のディスク装置からの読み出し、

（２）パリティグループの他のブロックの値の参照。

【００２３】本発明によって、上記の処理のうち、処理（２）が可能になり、しかも、シーケンシャル書き込みの場合には、他のブロックへの書き込み要求が既に行なわれているため、メモリ上で他のブロックの値が参照できる。すなわち、パリティデータの生成時に、ディスク

10

装置からの読み出しが発生しない。その結果、断片化されたファイルのシーケンシャル書き込み時に、書き込み処理が効率的に行える。

【００２４】

【実施例】以下、本発明の第一の実施例を詳細に説明する。本実施例の処理手順は、図１に示すように以下のプログラム(1)～(4)から構成される。

【００２５】(1)統合ファイル管理プログラム3201…分割されたデータ2001～2004とパリティデータ2010を管理するサブファイル管理テーブル4201～4204,4300のポインタを格納する統合ファイル管理テーブル4100が設けられ、この統合ファイル管理テーブル4100を用いてファイルが管理される。

【００２６】(2)サブファイル管理プログラム3301…ファイルストライピングプログラム3101により分割されたデータ2001～2004と、パリティデータ生成プログラム3401により生成されたパリティデータ2010とを格納する複数の二次記憶装置1001～1005内のブロックのブロック番号を二次記憶装置毎に管理するサブファイル管理テーブル4201～4204,4300が設けられ、サブファイル管理テーブルを用いてデータ2001～2004が管理される。

【００２７】(3)ファイルストライピングプログラム3101…ファイル2000のデータが、論理ブロックごとに複数のデータ2001～2004に分割される。

【００２８】(4)パリティデータ生成プログラム3401…ファイルストライピングプログラム3101により分割されたデータ2001～2004の先頭バイトから同一バイト変位ごとのデータの排他的論理和が、パリティデータ2010として格納される。

【００２９】また、ユーザは、以下のファイル操作コマンド(5)～(10)を使用してファイルアクセスを行う。ファイル操作コマンド(5)～(10)によって、プログラム(1)～(4)が起動され、以下の処理が行なわれる。

【００３０】(5)CREATEコマンド7100：ファイルが新規に作成される。

(6)WRITEコマンド7200：ファイルのデータが書き込まれる。

(7)OPENコマンド7300：ファイル操作を行うための統合ファイル管理テーブルが獲得される。

【００３１】(8)READコマンド7400：ファイルのデータが読み出される。

(9)CLOSEコマンド7500：CREATEコマンドまたはOPENコマンドで獲得した統合ファイル管理テーブルが解放される。

(10)RECOVERYコマンド7600：故障ディスク装置内のデータにアクセスされる。

【００３２】以下では、上記のプログラム(1)～(4)とファイル操作コマンド(5)～(10)の詳細を説明する。

(1)統合ファイル管理プログラム3201

統合ファイル管理プログラム3201の構成を図２に示す。

11

統合ファイル管理プログラムは、さらに以下の３つのサブプログラムから構成される。

【００３３】・統合ファイル管理テーブル割り当てプログラム3210…CREATEコマンド7100やOPEN7300コマンドの発行時に起動され、未使用の統合ファイル管理テーブル4100がCREATEコマンド7100やOPENコマンド7300の対象のファイル2000に割り当てられる。

【００３４】・統合ファイル管理テーブル検索プログラム3220…OPENコマンド7300の発行時に起動され、OPENコマンドで指定されたファイルの統合ファイル管理テーブル4100が検索される。

【００３５】・統合ファイル管理テーブル解放プログラム3230…CLOSEコマンド7500の発行時に起動され、CLOSEコマンドで指定されたファイルの統合ファイル管理テーブル4100が未使用テーブルとして解放される。

【００３６】また、統合ファイル管理プログラム3201で獲得、解放又は検索される統合ファイル管理テーブル4100は、図３に示すように、以下のエントリから構成される。

・ファイル名称エントリ4110…このエントリには、ユーザがファイルを一意に指定するための記述子が格納される。ユーザは、ファイルのOPEN時にファイル名称エントリに格納された記述子を用いてファイルを指定する。CREATEコマンドの発行時にファイル名称エントリに記述子が格納される。

【００３７】・第一～第四サブファイル管理テーブルへのポインタエントリ4121～4124…CREATEコマンドやOPENコマンドの発行時に、データ2001～2004を管理する各サブファイル管理テーブル4201～4204へのポインタが格納される。このエントリ4121～4124は、サブファイルの数だけ設けられる。

【００３８】・パリティ用サブファイル管理テーブルへのポインタ4130…パリティデータ2010を管理するサブファイル管理テーブル4300のポインタが格納される。第一～第四サブファイル管理テーブルへのポインタエントリ4121～4124と同じタイミングで、サブファイル管理テーブル4300へのポインタがこのエントリに格納される。

【００３９】統合ファイル管理テーブル割り当てプログラム3210は、図４に示すように三つのステップからなる。

(ステップ8210)統合ファイル管理テーブルのFREEリスト4150の中に、未使用の統合ファイル管理テーブルがあるかどうかが調べられる。未使用のテーブルがある場合にはステップ8212へ進み、未使用のテーブルがない場合にはステップ8211へ進む。

【００４０】(ステップ8211)統合ファイル管理テーブル解放プログラム3230のステップ8230（後述）によって、FREEリスト4150に統合ファイル管理テーブルが接続されるまでwaitする。

(ステップ8212)統合ファイル管理テーブル4100が、FREE

12

リスト4150から取り出されて、FREEリスト4150からUSEDリスト4160につなぎ替えられ、取り出された統合ファイル管理テーブル4100のポインタが、引き数として返される。

【００４１】統合ファイル管理テーブル検索プログラム3220は、図５に示すように４つのステップからなる。

(ステップ8220)ファイル名称をキーとして、USEDリスト4160に接続されている統合ファイル管理テーブル4100が検索される。

【００４２】(ステップ8221)該当する統合ファイル管理テーブル4100がある場合にはステップ8222に進み、該当するテーブルがない場合にはステップ8223に進む。

(ステップ8222)該当する統合ファイル管理テーブル4100のポインタが返される。　(ステップ8223)該当する統合ファイル管理テーブル4100が存在しないため、エラー値が返される。

【００４３】統合ファイル管理テーブル解放プログラム3230は、図６に示すように３つのステップからなる。

(ステップ8230)指定された統合ファイル管理テーブル4100のエントリが、すべて初期化されてUSEDリスト4160からFREEリスト4150に接続される。

【００４４】(ステップ8231)FREEリスト4150に統合ファイル管理テーブル4100が接続されるまでwaitしているプログラムがあるかどうかが調べられ、該当するプログラムがある場合にはステップ8232へ進み、該当するプログラムがない場合には処理を終了する。

(ステップ8232)waitしているプログラムのwait状態が解かれる。

【００４５】(2)サブファイル管理プログラム3301

サブファイル管理プログラム3301の構成を図７に示す。サブファイル管理プログラム3301は、さらに以下の４つのサブプログラムから構成される。

【００４６】・サブファイル管理テーブル割り当てプログラム3310…CREATEコマンド7100やOPENコマンド7300のコマンド発行時に起動され、未使用のサブファイル管理テーブル4200が、CREATEコマンド7100やOPENコマンド7300で指定されたファイル2000に割り当てられる。

【００４７】・サブファイル管理テーブル検索プログラム3320…OPENコマンド7300の発行時に起動され、OPENコマンドで指定されたファイルのサブファイル管理テーブル4200が検索されて取り出される。

【００４８】・サブファイル管理テーブル解放プログラム3330…CLOSEコマンド7500の発行時に起動され、CLOSEコマンドで指定されたファイルのサブファイル管理テーブル4200が未使用テーブルとして解放される。

【００４９】・データ格納位置決定プログラム3340…WRITEコマンド7200の発行時に起動され、ディスク装置上のどの位置にデータを格納するかが決定され、サブファイル管理テーブル4200に記録される。

【００５０】また、サブファイル管理プログラム3301で

13

獲得、解放又は検索されるサブファイル管理テーブル4200,4201～4204,4300は、図８に示すエントリから構成される。　・ファイル名称エントリ4250…ユーザがファイルを一意に指定するための記述子が格納される。CREATEコマンドの発行時に、ファイル名称エントリに記述子が格納される。

【００５１】・サブファイル管理テーブル番号エントリ4240…同一ディスク装置内の各サブファイル管理テーブルに付与される通し番号。

・ディスク装置番号エントリ4210…ディスク装置を一意に決定する番号。

・ブロック番号エントリ4220…データが実際に格納されるディスク装置の物理ブロック番号。

【００５２】・パリティファイルフラグエントリ4260…データがパリティデータであるか否かを示すフラグ。

・使用／未使用フラグ4270…サブファイル管理テーブルが使用されているかを示すフラグ。

【００５３】サブファイル管理テーブル4201～4204,4300は、ディスク装置1001～1005に保持されるだけでなく、ファイルがCREATEまたはOPENされた時にはメモリ上にキャッシングされる。

【００５４】サブファイル管理テーブル割り当てプログラム3301は、図９に示すように、５つのステップからなる。

(ステップ8310)ディスク装置のサブファイル管理テーブルエリア4250内にfreeなサブファイル管理テーブル4200があるかが調べられ、テーブルがある場合にはステップ8312へ進み、テーブルが無い場合にはステップ8311に進む。

【００５５】(ステップ8311)FREEなサブファイル管理テーブル4200ができるまでwaitする。

(ステップ8312)ディスク装置からFREEなサブファイル管理テーブル4200が取り出される。

【００５６】(ステップ8313)サブファイル管理テーブル4200の使用／未使用フラグ4270にONフラグが設定される。

(ステップ8314)サブファイル管理テーブル・キャッシュ4260にサブファイル管理テーブル4200がコピーされる。

【００５７】サブファイル管理テーブル検索プログラム3320は、図１０に示すように７つのステップからなる。

(ステップ8320)ファイル名称をキーにサブファイル管理テーブル・キャッシュ4260を検索する。

【００５８】(ステップ8321)該当するサブファイル管理テーブル4200がある場合にはステップ8325に進み、該当するテーブルがない場合にはステップ8322に進む。

(ステップ8322)ファイル名称をキーとして、ディスク装置上のサブファイル管理テーブルエリア4250が検索される。

【００５９】(ステップ8323)該当するサブファイル管理テーブル4200がサブファイル管理テーブルエリア4250内

14

にある場合にはステップ8324に進み、該当するテーブルが無い場合にはステップ8326に進む。

(ステップ8324)該当するサブファイル管理テーブルエリア4250内のサブファイル管理テーブル4200が、サブファイル管理テーブル・キャッシュ4260にコピーされる。

【００６０】(ステップ8325)該当するサブファイル管理テーブル4200のポインタが返される。　(ステップ8326)該当するサブファイル管理テーブル4200が存在しないため、エラー値が返される。

【００６１】サブファイル管理テーブル解放プログラム3330は、図１１に示すように４つのステップからなる。

(ステップ8330)指定されたサブファイル管理テーブル4200は、サブファイル管理テーブル・キャッシュ4260上で無効にされる。

(ステップ8331)指定されたサブファイル管理テーブル4200は、ディスク装置上のサブファイル管理テーブルエリア4250上で無効にされる。

【００６２】(ステップ8332)FREEなサブファイル管理テーブル4200が接続されるまでwaitしているプログラムがあるかどうかが調べられ、該当するプログラムがある場合にはステップ8333に進み、該当するプログラムがない場合には処理を終了する。

(ステップ8333)該当するプログラムのwait状態が解かれる。

【００６３】データ格納位置決定プログラム3340は、図１２に示すように、次の二つのステップからなる。

(ステップ8340)ディスク装置のデータ領域内の未使用ブロックがファイルデータ格納領域として割り当てられる。

(ステップ8341)サブファイル管理テーブル4200のブロック番号エントリ4220に割り当てられたブロック番号が登録される。

【００６４】(3)ファイルストライピングプログラム3101

ファイルストライピングプログラム3101は、図１３に示すように、次の３つのステップからなる。

【００６５】(ステップ8101)ファイルストライピングプログラムを起動するプログラムから渡されたファイル論理ブロック番号とディスク装置台数とから、サブファイル論理ブロック番号が計算される。

【００６６】(ステップ8102)ステップ8101と同様に、データを格納するディスク装置のディスク装置番号が計算される。

(ステップ8103)ステップ8101とステップ8102とにより計算されたサブファイル論理ブロック番号とディスク装置番号とが返される。

【００６７】(4)パリティ生成プログラム3401

パリティ生成プログラム3401は、図１４に示す次の５つのステップからなる。

(ステップ8401)呼び出し元プログラムから渡された統合

15

ファイル管理テーブル4100へのポインタに基づいて、統合ファイル管理テーブル4100が獲得される。

【００６８】(ステップ8402)獲得された統合ファイル管理テーブル4100に対応するサブファイル管理テーブルへのポインタエントリ4121～4124,4130を参照して、サブファイル管理テーブルが獲得される。

【００６９】(ステップ8403)パリティデータ格納用のバッファが確保される。

(ステップ8404)呼び出し元プログラムから渡されたパリティグループ番号に対応するパリティグループのブロックに基づいて、パリティデータが生成され、パリティデータ格納用のバッファに書き込まれる。

(ステップ8405)パリティデータ格納用のバッファのポインタが返される。

【００７０】ユーザに提供されるファイル操作コマンド(5)～(10)は、上記のプログラム(1)～(4)を組み合わせることによって実現される。ファイル操作コマンド(5)～(10)の外部仕様を図１５に示す。以下、各コマンドによって実行される処理の詳細フローを示す。

【００７１】(5)CREATEコマンド7100

CREATEコマンド7100の実行の際に、ファイル名称が引数としてCREATEコマンド7100に渡される。CREATEコマンド7100が実行されると、戻り値として統合ファイル管理テーブルのポインタが返される。CREATEコマンド7100に対応する処理は、図１６に示す次の６つのステップからなる。

【００７２】(ステップ9100)統合ファイル管理プログラム3201が起動されて、統合ファイル管理テーブル4100が新規に獲得される。

(ステップ9101)サブファイル管理プログラム3101が起動されて、サブファイル管理テーブル4201～4204,4300がディスク装置の数だけ新規に獲得される。

【００７３】(ステップ9102)所定の基準に基づいて、パリティデータ格納ディスク装置が決定され、そのディスク装置に対応するサブファイル管理テーブル4300へのポインタが統合ファイル管理テーブル4100に含まれるパリティ用サブファイル管理テーブルへのポインタエントリ4130に登録され、そのサブファイル管理テーブル4300のパリティファイルフラグがON状態に設定される。なお、パリティデータ格納ディスク装置をすべてのファイルで共通にしてもよいし、ファイルごとにパリティデータ格納ディスク装置を変更してもよい。各ファイルごとにパリティデータ格納ディスク装置を変更する場合には、各ディスク装置の使用量の中で、最も使用量の少ないディスク装置をパリティデータ格納ディスク装置に割り当てる方法、各ディスク装置へのアクセス頻度の統計に基づいてパリティデータを格納すべきディスク装置を決定する方法を用いることができる。また、ファイル作成順に順番にパリティデータ格納ディスク装置を移動させていく方法も用いることができる。

16

【００７４】(ステップ9103)上記以外のサブファイル管理テーブル4201～4204は、第一～第四サブファイル管理テーブルへのポインタエントリ4121～4124に登録される。

(ステップ9104)ユーザが指示したファイル名称が、統合ファイル管理テーブル4100と各サブファイル管理テーブル4121～4124,4300のファイル名称エントリに登録される。

(ステップ9105)統合ファイル管理テーブル4100のポインタが返される。

【００７５】(6)WRITEコマンド7200

WRITEコマンド7200の実行時には、引数として、統合ファイル管理テーブル4100のポインタ、ファイルのどの位置にデータを書き込むかを表すファイルオフセット値、書き込みデータサイズ、及び書き込み元のデータが格納されているユーザ空間上のデータバッファが、WRITEコマンド7200に渡される。WRITEコマンド7200は戻り値を返さない。

【００７６】WRITEコマンド7200は、図１７に示すように、次の１２のステップからなる。

(ステップ9200)ユーザが指示した統合ファイル管理テーブル4100のポインタに基づいて、統合ファイル管理テーブル4100が獲得される。

【００７７】(ステップ9201)ユーザが指示したファイルオフセット値に基づいて、ファイルの論理ブロック番号が計算される。

(ステップ9202)ユーザが指示した書き込みサイズに基づいて、書き込みに必要な論理ブロックの個数が計算される。

(ステップ9203)処理すべき論理ブロック数として、ステップ9202で求めた論理ブロックの個数が設定される。

【００７８】(ステップ9204)書き込みデータ用のバッファが設けられ、ユーザ空間上の書き込み対象データがバッファにコピーされる。

(ステップ9205)論理ブロック番号を引数としてファイルストライピングプログラム3101が起動され、サブファイルの論理ブロック番号とデータとを格納するディスク装置が決定される。

【００７９】(ステップ9206)統合ファイル管理テーブル4100へのポインタとステップ9204で求めたサブファイルの論理ブロック番号（パリティグループ番号）とを引数として、パリティ生成プログラム3401が起動される。

(ステップ9207)統合ファイル管理テーブル4100に基づいて、データを格納すべきディスク装置とパリティ用のサブファイル管理テーブル4200とが獲得される。

【００８０】(ステップ9208)データ格納用とパリティ用のサブファイル管理テーブル4200を引数として、サブファイル管理プログラム3301に含まれるデータ格納位置決定プログラム3340が起動され、ディスク装置のどの位置にデータを格納するかが決定される。

(ステップ9209)データとパリティデータとがディスク装置に書き込まれる。

【００８１】(ステップ9210)処理すべきブロック数がディクリメントされ、論理ブロック番号がインクリメントされる。

(ステップ9211)処理すべきブロック数が正である場合にはステップ9205に戻り、ブロック数が０以下となった場合には処理を終了する。

【００８２】(7)OPENコマンド7300

OPENコマンド7300の実行時には、ファイル名称が引数としてOPENコマンド７３００に渡される。ＯＰＥＮコマンド7300は、処理が実行されると、戻り値として統合ファイル管理テーブルのポインタを返す。OPENコマンド7300は、図１８に示すように、次の１１のステップからなる。

【００８３】(ステップ9300)ユーザが指示したファイル名称を引数として、統合ファイル管理プログラム3201に含まれる統合ファイル管理テーブル検索プログラム3320が起動される。

(ステップ9301)統合ファイル管理プログラム3201からの戻り値がエラー値であるかどうかが調べられ、戻り値がエラー値である場合にはステップ9302に進み、戻り値がエラー値でなく統合ファイル管理テーブル4100へのポインタである場合にはステップ9309に進む。

【００８４】(ステップ9302)検索ディスク数がディスク台数に格納される。

(ステップ9303)ユーザが指示したファイル名称を引数として、サブファイル管理プログラム3301に含まれるサブファイル管理テーブル検索プログラム3320が起動される。

【００８５】(ステップ9304)検索ディスク数がディクリメントされる。

(ステップ9305)検索ディスク数が正の値である場合にはステップ9303に進み、検索ディスク数が０以下になった場合にはステップ9306に進む。

【００８６】(ステップ9306)サブファイル管理プログラム3301からの戻り値がエラー値であるかどうかがチェックされ、戻り値がエラー値である場合にはステップ9310に進み、戻り値がエラー値でなくサブファイル管理テーブル4200のポインタである場合にはステップ9307に進む。

【００８７】(ステップ9307)統合ファイル管理プログラム3201に含まれる統合ファイル管理テーブル割り当てプログラム3210が起動され、統合ファイル管理テーブル4100が新規に獲得される。

(ステップ9308)統合ファイル管理テーブル4100に、ファイル名称と検索したサブファイル管理テーブル4200のポインタとが登録される。

(ステップ9309)統合ファイル管理テーブル4100のポインタが返されて処理を終了する。

(ステップ9310)エラー値が返されて処理を終了する。

【００８８】(8)READコマンド7400

READコマンド7400の実行時には、統合ファイル管理テーブル4100のポインタ、ファイルのどの位置を読み出すかを表すファイルオフセット値、読み出しデータサイズ、及び読み出したデータを格納するユーザ空間上のデータバッファとが引数としてREADコマンドに渡される。READコマンドは、実行した結果を戻り値として返さない。

【００８９】READコマンドは、図１９に示すように次の１２のステップからなる。

(ステップ9400)ユーザが指定した統合ファイル管理テーブルのポインタに基づいて統合ファイル管理テーブルが獲得される。

(ステップ9401)ユーザが指定したファイルオフセット値に基づいてファイルの論理ブロック番号が計算される。

(ステップ9402)ユーザが指定した読み出しサイズに基づいて論理ブロックの個数が計算される。

【００９０】(ステップ9403)読み出しデータ用のバッファが確保される。

(ステップ9404)処理すべき論理ブロック数が論理ブロックの個数に設定される。 (ステップ9405)ファイルの論理ブロック番号を引数としてファイルストライピングプログラム3101が起動され、読み出すデータが格納されているサブファイルの論理ブロック番号とデータが格納されているディスク装置とが決定される。

【００９１】(ステップ9406)統合ファイル管理テーブル4100に基づいて、データを読み出すディスク装置のサブファイル管理テーブル4200が獲得される。

(ステップ9407)サブファイル管理テーブル4200のブロック番号エントリ4220を参照して、ディスク装置のどの位置にデータが格納されているかが決定される。

(ステップ9408)データがディスク装置から読み出される。

【００９２】(ステップ9409)処理すべきブロック数がディクリメントされ、論理ブロック番号がインクリメントされる。

(ステップ9410)処理すべきブロック数が正の値である場合にはステップ9405に進み、ブロック数が０以下になった場合にはステップ9411に進む。

(ステップ9411)読み出したデータがユーザ空間にコピーされる。

【００９３】(9)CLOSEコマンド7500

CLOSEコマンド7500の実行時には、統合ファイル管理テーブル4100へのポインタが引数としてCLOSEコマンド7500に渡される。CCLOSEコマンド7500は、実行後に戻り値を返さない。

【００９４】CLOSEコマンド7500は、図２０に示すように７つのステップからなる。

(ステップ9500)ユーザが指定した統合ファイル管理テーブル4100のポインタに基づいて統合ファイル管理テーブ

19

ル4100が獲得される。
(ステップ9501)統合ファイル管理テーブル4100に基づいて全ディスク装置のサブファイル管理テーブル4201～4204,4300が獲得される。
【００９５】(ステップ9502)解放すべきサブファイル数がディスク装置台数に設定される。
(ステップ9503)サブファイル管理テーブルのポインタを引数としてサブファイル管理プログラム3301に含まれるサブファイル管理テーブル解放プログラム3330が起動され、サブファイル管理テーブルが無効にされる。
(ステップ9504)解放すべきサブファイル数がディクリメントされる。
【００９６】(ステップ9505)解放すべきサブファイル数が正の値である場合にはステップ9503に戻り、サブファイル数が０以下の場合にはステップ9506に進む。
(ステップ9506)統合ファイル管理テーブル4100のポインタを引数として、統合ファイル管理プログラム3201に含まれる統合ファイル管理テーブル解放プログラム3230が起動され、統合ファイル管理テーブル4100が解放される。
【００９７】(10)RECOVERYコマンド7600
RECOVERYコマンド7600の実行時には、統合ファイル管理テーブルのポインタがRECOVERYコマンド7600に渡される。RERCOVERYコマンドは、実行後に戻り値を返さない。
【００９８】RECOVERYコマンド7600は、図２１に示すように、次の８つのステップからなる。 (ステップ9600)ユーザが指定した統合ファイル管理テーブルのポインタに基づいて統合ファイル管理テーブルが獲得される。
(ステップ9601)故障していない全てのディスク装置に対応するサブファイル管理テーブルが、統合ファイル管理テーブルのサブファイル管理テーブルエントリを参照して獲得される。
【００９９】(ステップ9602)復元すべきサブファイルのデータブロック数がファイルサイズとディスク台数とから計算される。
(ステップ9603)復元すべき論理ブロック番号が０に初期化される。
【０１００】(ステップ9604)故障していない全てのディスク装置に対応するサブファイルから復元すべき論理ブロック番号が読み出される。
(ステップ9605)パリティ生成プログラム3401が起動され、故障したディスク装置に対応するサブファイルに含まれる該当論理ブロック番号のブロックが復元される。
【０１０１】(ステップ9606)復元すべき論理ブロック番号がインクリメントされ、復元すべきデータブロック数がディクリメントされる。
(ステップ9607)復元すべきデータブロック数が正の値の場合にはステップ９６０４に戻り、データブロック数が０以下の場合には処理を終了する。

20

【０１０２】上記のファイル操作コマンド（５）～(10)を組み合わせて、ユーザはファイルアクセスを行う。一例として、ファイル書き込み、ファイル読み出し、およびディスク装置故障時のファイル回復の３つの操作をユーザが行う際の手順を示す。
【０１０３】・ファイル書き込み時のユーザ手順
ファイル書き込み時のユーザ手順の一例を図２２に示す。
(ステップ10100)ユーザは、ファイルを新規に作成するか、それとも既に存在するファイルに上書きするかを判断し、新規に作成する場合にはステップ10101に進み、上書きする場合にはステップ10102に進む。
(ステップ10101)ユーザは、CREATEコマンド7100を発行する。
(ステップ10102)ユーザは、OPENコマンド7200を発行する。
【０１０４】(ステップ10103)CREATEコマンド7100またはOPENコマンド7300によって獲得された統合ファイル管理テーブル4100のポインタを用いて、ユーザは、WRITEコマンド7200を発行する。
(ステップ10104)WRITEコマンド7200の処理が終了した後、ユーザは、CLOSEコマンド7500を発行する。
【０１０５】・ファイル読み出し時のユーザ手順
ファイル読み出し時のユーザ手順の一例を図２３に示す。
(ステップ10200)ユーザは、OPENコマンド7300を発行する。
(ステップ10201)OPENコマンド7300によって獲得された統合ファイル管理テーブルのポインタを用いて、ユーザは、READコマンド7400を発行する。
(ステップ10202)READコマンド7400の処理が終了した後、ユーザは、CLOSEコマンド7500を発行する。
【０１０６】・ファイル回復時のユーザ手順
ファイル回復時のユーザ手順の一例を図２４に示す。
(ステップ10300)ユーザは、OPENコマンド7300を発行する。
(ステップ10301)OPENコマンド7300によって獲得された統合ファイル管理テーブルのポインタを用いて、ユーザは、RECOVERコマンド7600を発行する。
(ステップ10302)ユーザは、CLOSEコマンドを発行する。
【０１０７】上記のように、ユーザは、本実施例で示した操作コマンドを用いて、ファイルの書き込み、読み出し、あるいはディスク装置故障時のファイル回復を行うことができる。特に、ファイル書き込み時に、ファイルのデータをディスク装置の連続領域に書き込めずに断片化されてしまった場合でも、パリティ生成を効率的に行なうことができる。
【０１０８】図２５は、D.A.Patterson, et al. "Introduction to Redundant Arrays of Inexpensive Disks(RAID)", spring COMPCON'89, pp.112-117, Feb.1989に示

されている従来のRAIDアーキテクチャによるパリティ生成と、本実施例のパリティ生成とを比較した結果を示す。図２５に示すように、ファイルがディスク装置上に格納された場合、ファイルの先頭から８ブロックのデータが書き込まれると、従来のRAIDでは６ブロックのパリティデータを書き込む必要がある。一方、本実施例では、２ブロックのパリティデータを書き込むだけでよい。パリティデータの生成のための前処理では、ファイル書き込みの際に発生するブロックI/Oの数は以下のようになる。

【０１０９】従来のRAIDでは、パリティ生成のための前処理で読み出されるブロック数が１２ブロック、書き込みデータのブロック数が８ブロック、さらに、パリティデータのブロック数が６ブロックであり、計２６ブロックのブロックI/Oが発生する。本実施例では、パリティ生成のための前処理で読み出されるブロック数は０ブロック、書き込みデータのブロック数は８ブロック、さらに、パリティデータのブロック数は２ブロックであり、計１０ブロックのブロックI/Oが発生する。すなわち、本実施例では、従来のRAIDに比べてファイル書き込みの際のパリティ生成処理が効率よく行われ、ファイル書き込み性能が向上する。

【０１１０】次に、本発明の第二の実施例について詳細に説明する。第一の実施例との差異は、パリティデータの格納の仕方にある。すなわち、第一の実施例では、一つのファイルのパリティデータは、ディスク装置1001～1005のうちの一つのディスク装置に格納されていたが、第二の実施例では、一つのファイルのパリティデータが複数のディスク装置1001～1005に分散されて格納される。

【０１１１】第一の実施例では、パリティデータを格納するディスク装置がファイル毎に変更されるので、複数のファイルへの同時書き込みの際に、パリティデータの書き込みに伴う負荷が複数のディスク装置に分散される。しかし、一つのファイルに対する複数の書き込み要求が同時に発生した場合には、パリティデータの書き込みに伴う負荷が一つのディスク装置に集中してしまう。本実施例では、一つのファイルに格納されていたパリティデータを複数のディスク装置1001～1005に分散して格納することにより、上記の問題を解決している。

【０１１２】図２６は第二の実施例の動作原理を示す。第二の実施例では、第一の実施例のCREATEコマンド7100、ファイルストライピングプログラム3101、及び統合ファイル管理テーブル4100が変更される。CREATEコマンドの発行時に、第一の実施例の図１６のステップ9102で行なわれていた、パリティデータ格納ディスク装置を決定する処理の代わりに、第二の実施例では、パリティデータの書き込み開始ディスク装置を決定する処理が行なわれる。そして、決定したパリティデータの書き込み開始ディスク装置に対応するサブファイル管理テーブルに

含まれるパリティファイルフラグエントリ4260のフラグがON(1)に設定される。ファイルストライピングプログラムは、このパリティファイルフラグエントリ4260のフラグを参照して、フラグがONになっているサブファイルの１ブロック目、次のサブファイルの２ブロック目、その次のサブファイルの３ブロック目、のようにパリティデータが格納されているブロックの有無を順次判断し、読み出し要求や書き込み要求がなされているファイルデータがどのブロックにあるかを計算する。

【０１１３】本実施例で使用するCREATE7101コマンドとファイルストライピングプログラム3102を詳細に説明する。

・CREATEコマンド7101

CREATEコマンド7101の詳細フローを図２７に示す。(ステップ9100)～(ステップ9101)、及び(ステップ9103)～(ステップ9105)までは第一の実施例のCREATEコマンド7101と同じであり、第一の実施例の(ステップ9102)だけが(ステップ9106)に置き換わる。以下では、(ステップ9106)のみを説明する。

【０１１４】(ステップ9106)パリティデータの書き込み開始ディスク装置が決定され、そのディスク装置に対応するサブファイル管理テーブルのパリティファイルフラグがONに設定される。この場合、パリティデータの書き込み開始ディスク装置は、ディスク装置1001～1005のうち、どれか一つのディスク装置に固定してもよい。また、ファイルの作成順に書き込み開始ディスク装置を順次巡回させても良い。また、ディスク装置の使用量やアクセス頻度に基づいて、パリティデータの書き込み開始ディスク装置を決定しても良い。

【０１１５】・ファイルストライピングプログラム3102

ファイルストライピングプログラム3102の詳細フローを図２８に示す。第二に実施例では、第一の実施例のファイルストライピングプログラム3101の(ステップ8102)と(ステップ8103)の間に、(ステップ8104)～(ステップ8106)の３ステップを付け加えられている。以下では、追加された(ステップ8104)～(ステップ8106)のみを説明する。

【０１１６】(ステップ8104)サブファイル論理ブロック番号、ディスク装置台数、およびサブファイル管理テーブルのパリティフラグで識別されるパリティ格納開始ディスク装置番号に基づいてパリティディスク装置番号が計算される。

(ステップ8105)パリティディスク装置番号とディスク装置番号とが比較され、パリティディスク装置番号がディスク装置番号よりも大きければステップ8103に進み、そうでなければステップ8106に進む。

(ステップ8106)ディスク装置番号がインクリメントされる。

【０１１７】第一の実施例では、パリティデータを格納すべきディスク装置をファイル毎に変更されるので、複

数のファイルへの同時書き込みの際に、パリティデータの書き込みに伴う負荷が複数のディスク装置に分散される。しかし、一つのファイルに対する複数の書き込み要求が同時に発生した場合、パリティデータの書き込みに伴う負荷が一つのディスク装置に集中してしまう。本実施例では、一つのファイルのパリティデータがディスク装置1001～1005に分散して格納されるので、上記の問題が解決される。

【０１１８】以上のように、サブファイル管理テーブル4200、CREATEコマンド7100、およびファイルストライピングプログラム3101を変更することにより、パリティータがディスク装置1001～1005に分散して格納される。その結果、一つのファイルへの複数の書き込み要求が発生した場合、パリティデータの書き込みに伴う負荷が一つのディスク装置に集中してしまうという第一の実施例の問題点が解決される。

【０１１９】次に、第三の実施例について詳細に説明する。第一及び第二の実施例では、ファイルデータ書き込み時に、ファイルデータの書き込みと同期をとってパリティデータが作成されていた。しかし、第一及び第二の実施例では、同一のブロックに対して複数の書き込み要求が次々と発生するような場合に、パリティデータを何回も生成する必要がある。そのため、上記の場合には、パリティデータ生成がボトルネックになる可能性がある。そこで、本実施例では、ファイルデータの書き込みとパリティデータの書き込みとを非同期に実行することによって、上記の問題を解決する。

【０１２０】本実施例では、図２９に示すように、第一及び第二の実施例の構成に、バッファキャッシュ4500、パリティグループ管理テーブル4600、及びバッファキャッシュ管理プログラム3600が付加される。

【０１２１】バッファキャッシュ4500は、パリティグループ管理テーブルへのポインタエントリ4520を持つバッファヘッダ4510とデータ格納エリア2001からなる。また、パリティグループ管理テーブル4600は、統合ファイル管理テーブルへのポインタエントリ4610と、バッファへのポインタエントリ4621～4625からなる。WRITEコマンド7200における書き込み用バッファ獲得処理ステップ9204、及びREADコマンド7400における読み出し用バッファ獲得処理ステップ9403では、バッファキャッシュ管理プログラム３６００を介してバッファ獲得が行なわれるように、処理内容が変更される。

【０１２２】図３０に、バッファキャッシュ管理プログラム3600の詳細フローを示す。バッファキャッシュ管理プログラム3600は、次の14ステップからなる。

(ステップ8600)呼び出し元のプログラムから受け取ったファイルの論理ブロック番号に基づいて、パリティグループ番号が計算される。

(ステップ8601)呼び出し元のプログラムから受け取った統合ファイル管理テーブル4100のポインタと、ステップ

8600で計算したパリティグループ番号とに基づいてパリティグループ管理テーブル全体が検索される。

【０１２３】(ステップ8602)該当するパリティグループ管理テーブルがある場合にはステップ8603へ進み、テーブルがない場合にはステップ8604へ進む。

(ステップ8603)該当するパリティグループ管理テーブル4600が獲得され、ステップ8605へ進む。

(ステップ8604)新規にパリティグループ管理テーブル4600が割り当てられる。

【０１２４】(ステップ8605)ファイルの論理ブロック番号を引数としてファイルストライピングプログラム3101が起動され、サブファイルの論理ブロック番号とデータを格納するディスク装置とが決定される。

(ステップ8606)獲得されたパリティグループ管理テーブルに、該当するサブファイルのバッファキャッシュ4500が接続されているかが調べられ、キャッシュが接続されている場合にはステップ8607に進み、キャッシュが接続されていない場合には8608に進む。

【０１２５】(ステップ8607)該当するバッファキャッシュ4500が獲得され、処理を終了する。 (ステップ8608)未使用のバッファキャッシュがあるかどうかが調べられ、未使用のキャッシュがある場合にはステップ8609に進み、未使用のキャッシュがない場合にはステップ8610に進む。

(ステップ8609)バッファキャッシュ4500が、新規に割り当てられ、パリティグループ管理テーブル4600に接続されて処理を終了する。

【０１２６】(ステップ8610)Least Recently Used Algorithm(LRU)アルゴリズムに従って、不要なバッファキャッシュ4500が選択される。本実施例では、LRUアルゴリズムを使用しているが、First In First Out (FIFO)アルゴリズムなど他のアルゴリズムを使用することも可能である。

【０１２７】(ステップ8611)選択されたバッファキャッシュ4500のパリティデータが計算されているかどうかが調べられ、パリティデータが計算されている場合にはステップ8613に進み、パリティデータが計算されていない場合にはステップ8612に進む。 (ステップ8612)パリティ生成プログラム3401が起動され、パリティ生成が行なわれる。

【０１２８】(ステップ8613)選択されたバッファキャッシュ4500のデータがディスク装置に書き込まれ、その後、バッファキャッシュ4500が未使用状態にされ、ステップ8608に進む。

【０１２９】バッファキャッシュ管理プログラム3600を使用したWRITEコマンド7201の処理フローを図３１に示し、READコマンド7401の処理フローを図３２に示す。

・WRITEコマンド７２０１

本実施例のＷＲＩＴＥコマンド7201は、第一の実施例のWRITEコマンド7200とは異なり、ディスク装置にデータ

を書き込まない。図３１に示すステップ9200～9202は第一の実施例と同じである。以下では、ステップ9202以降の処理を説明する。

【０１３０】(ステップ9208)データ格納用とパリティ用のそれぞれのサブファイル管理テーブルを引数として、サブファイル管理プログラム3301に含まれるデータ格納位置決定プログラム3340が起動され、ディスク装置のどの位置にデータを格納するかが決定される。

【０１３１】(ステップ9212)バッファキャッシュ管理プログラムが起動され、バッファキャッシュが獲得される。

(ステップ9213)獲得したバッファキャッシュに、ユーザ空間上の書き込み対象データがコピーされて処理を終了する。

【０１３２】・READコマンド7401

本実施例のREADコマンド7401は、第一の実施例のREADコマンド7400とは異なり、バッファキャッシュ上にデータが保持されているブロックについては、ディスク装置からデータを読み出さず、バッファキャッシュ上のデータを用いる。そのため、図３２では、図１９に示したステップ9403が除かれ、ステップ9412とステップ9413とがステップ9404とステップ9405との間に挿入される。以下では、ステップ9412とステップ9413について説明する。

【０１３３】(ステップ9412)バッファキャッシュ管理プログラム3600が起動される。

(ステップ9413)該当バッファキャッシュが獲得されたか、すなわちキャッシュがヒットしたかどうかが調べられ、キャッシュがヒットした場合にはステップ9409に進み、キャッシュがヒットしない場合にはステップ9405に進む。

【０１３４】上記の方法により、ファイルデータの書き込みとパリティ生成とが非同期で実行される。また、ディスクキャッシュ4500にデータをキャッシングすることにより、ディスク装置1001～1005とのアクセス回数を低減できる。

【０１３５】次に、本発明の第四の実施例を図３３に示し、詳細に説明する。本実施例では、高速にファイルのオープン処理を行えるようにするために、各サブファイル管理テーブル4201～4205に、それぞれディスクに隣接するディスクのサブファイル管理テーブルのサブファイル管理テーブル番号を格納するエリア4260が設けられている。

【０１３６】第一の実施例では、ユーザが指定したファイル名称に対応する統合ファイル管理テーブルが無いときに、各ディスク装置1001～1005からユーザの指定したファイル名称で表されるサブファイル管理テーブルが検索される。その場合には、ディスク装置1001～1005にある全てのサブファイル管理テーブルを読み出して検索する必要があり、処理に時間がかかる。

【０１３７】本実施例では、隣接ディスクのサブファイル管理テーブルのサブファイル管理テーブル番号4260がファイル作成時にサブファイル管理テーブルに記録される。そして、ファイルオープン時には、ディスク装置1001内のサブファイル管理テーブル4201だけがファイル名称によって検索されて獲得される。他のディスク装置1002～1005のサブファイル管理テーブル4202～4205は、それぞれ隣接ディスクのサブファイル管理テーブル番号を参照することによって獲得され、ファイル名称による検索が不要となる。

【０１３８】本実施例のOPENコマンド7301の処理内容を図３４を用いて説明する。本実施例のOPENコマンド7201では、図１７に示す第一の実施例のOPENコマンド7200におけるステップ9304～ステップ9306の代わりに、ステップ9311からステップ9314が実行される。以下では、ステップ9311～ステップ9314の処理を説明する。

【０１３９】(ステップ9311)サブファイル管理プログラムからの戻り値がエラー値であるか否かが判定され、戻り値がエラー値である場合にはステップ9310に進み、そうでない場合にはステップ9312に進む。

(ステップ9312)獲得したサブファイル管理テーブルの隣接ディスクサブファイル管理テーブル番号エントリを参照して、隣接ディスクのサブファイル管理テーブルが獲得される。

【０１４０】(ステップ9313)検索ディスク数がディクリメントされる。

(ステップ9314)検索ディスク数が正の値かどうかが調べられ、検索ディスク数が正の値であればステップ9312に進み、そうでなければステップ9307に進む。

【０１４１】上記の処理により、ファイル名称で検索されるサブファイル管理テーブルはテーブル4201の一つだけですむようになり、高速なファイルオープン処理が実現できる。

【０１４２】

【発明の効果】本発明によれば、従来のRAIDの各ディスク装置のブロックアドレスごとに排他的論理和を取る方法とは異なり、分割されたファイルの先頭位置からのバイト変位ごとに排他的論理和が取られる。そのため、分割されたサブファイルが各ディスク装置内の異なるアドレスのブロックに格納されても、分割されたサブファイル間でパリティデータを生成でき、書き込み性能の低下を防止できる。従って、高速なファイルアクセスが可能な高信頼ファイルシステムを低コストで実現できる。

【図面の簡単な説明】

【図１】第一の実施例の構成を表わした図である。

【図２】第一の実施例の統合ファイル管理プログラムの構成を表した図である。

【図３】第一の実施例における統合ファイル管理テーブルの構成を表わした図である。

【図４】第一の実施例における統合ファイル管理テーブル割り当てプログラムの処理を表した図である。

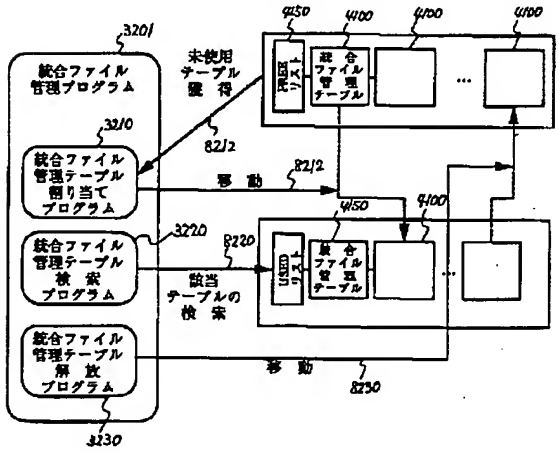
27

【図５】第一の実施例における統合ファイル管理テーブル検索プログラムの処理を表した図である。
【図６】第一の実施例における統合ファイル管理テーブル解放プログラムの処理を表した図である。
【図７】第一の実施例におけるサブファイル管理プログラムの構成を表した図である。
【図８】第一の実施例におけるサブファイル管理テーブルの構成を表した図である。
【図９】第一の実施例におけるサブファイル管理テーブル割り当てプログラムの処理を表した図である。
【図１０】第一の実施例におけるサブファイル管理テーブル検索プログラムの処理を表した図である。
【図１１】第一の実施例におけるサブファイル管理テーブル解放プログラムの処理を表した図である。
【図１２】第一の実施例におけるデータ格納位置決定プログラムの処理を表した図である。
【図１３】第一の実施例におけるファイルストライピングプログラムの処理を表した図である。
【図１４】第一の実施例におけるパリティ生成プログラムの処理を表した図である。
【図１５】第一の実施例におけるファイル操作コマンドのユーザインタフェースを表した図である。
【図１６】第一の実施例におけるCREATEコマンドの処理を表した図である。
【図１７】第一の実施例におけるWRITEコマンドの処理を表した図である。
【図１８】第一の実施例におけるOPENコマンドの処理を表した図である。
【図１９】第一の実施例におけるREADコマンドの処理を表した図である。

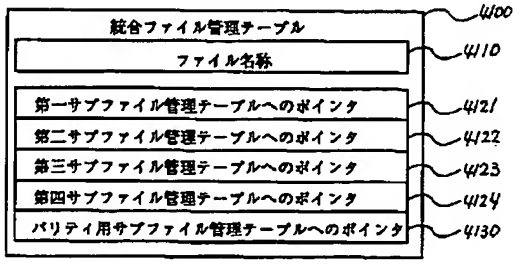
28

【図２０】第一の実施例におけるCLOSEコマンドの処理を表した図である。
【図２１】第一の実施例におけるRECOVERYコマンドの処理を表した図である。
【図２２】第一の実施例におけるユーザの書き込み処理手順を表した図である。
【図２３】第一の実施例におけるユーザの読み出し処理手順を表した図である。
【図２４】第一の実施例におけるユーザのファイル回復処理手順を表した図である。
【図２５】従来技術と第一の実施例のパリティ生成の比較を示した図である。
【図２６】第二の実施例の構成を表した図である。
【図２７】第二の実施例のCREATEコマンドの処理を表した図である。
【図２８】第二の実施例のファイルストライピングプログラムの処理を表した図である。
【図２９】第三の実施例の構成を表した図である。
【図３０】第三の実施例におけるバッファキャッシュ管理プログラムの処理を表した図である。
【図３１】第三の実施例におけるWRITEコマンドの処理を表した図である。
【図３２】第三の実施例におけるREADコマンドの処理を表した図である。
【図３３】第四の実施例の構成を表した図である。
【図３４】第四の実施例におけるOPENコマンドの処理を表した図である。

【図2】

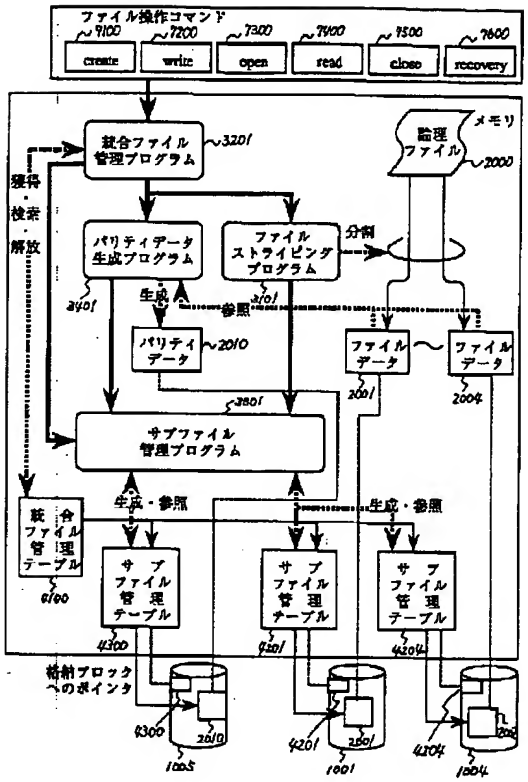


【図3】



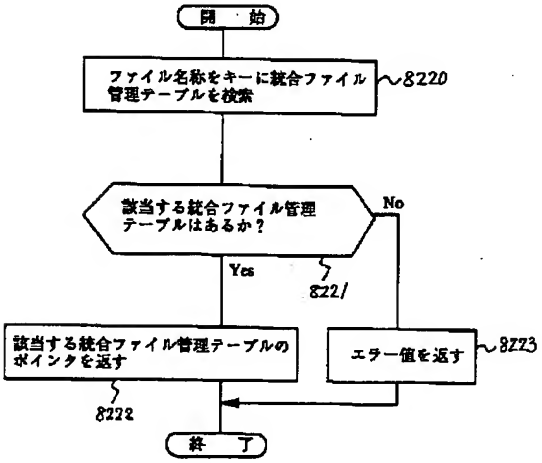
【図1】

図1



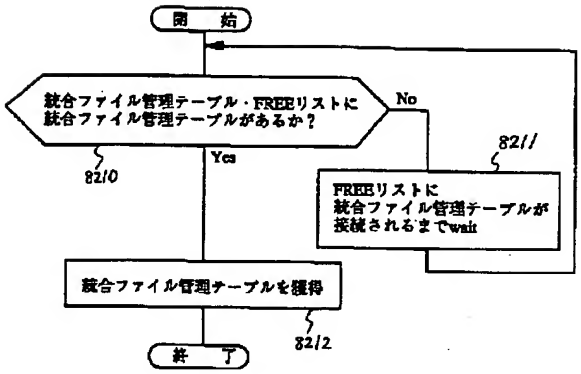
【図5】

図5 3220



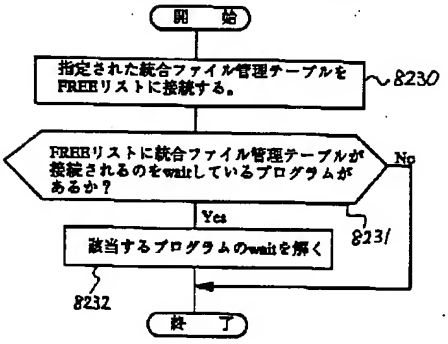
【図4】

図4 3210



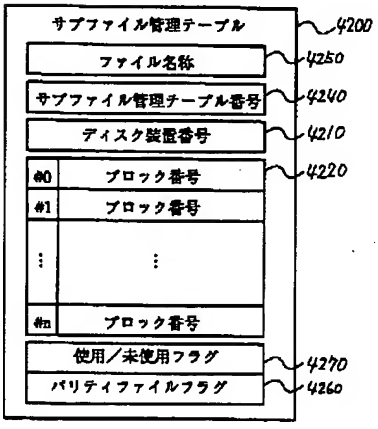
【図6】

図6 3230



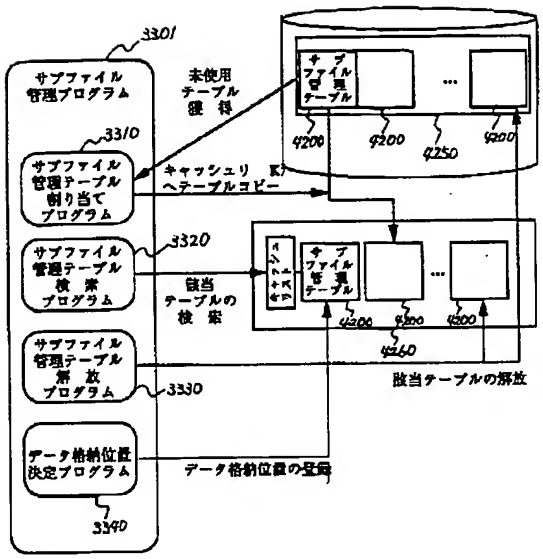
【図8】

図8



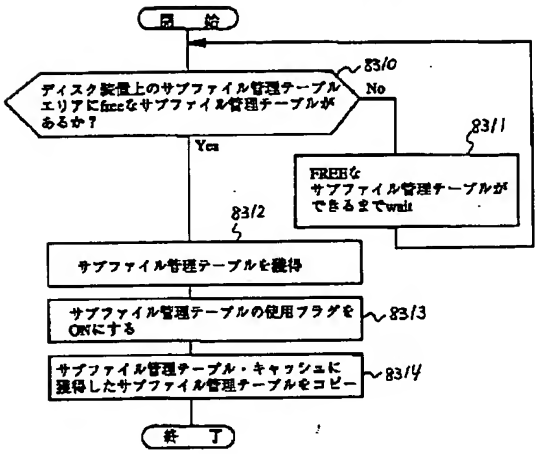
【図７】

図７



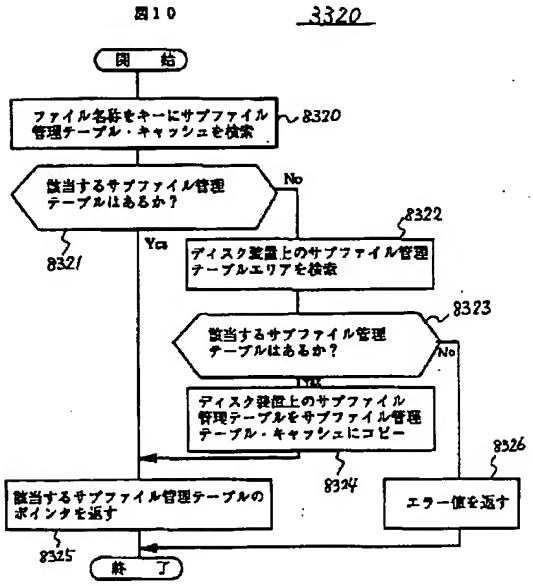
【図９】

図９　3301



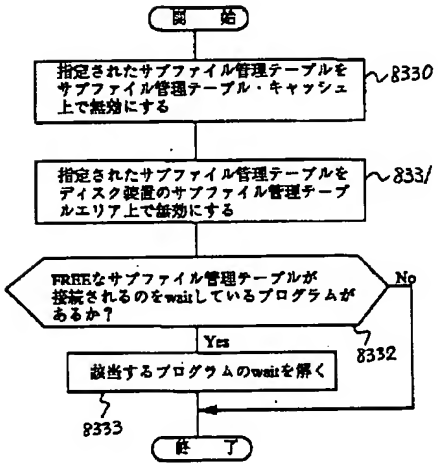
【図１０】

図１０　3320



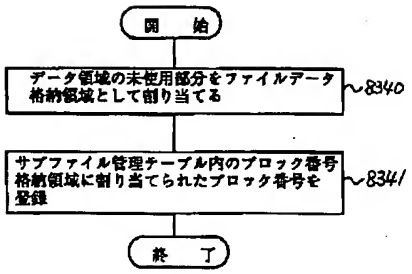
【図１１】

図１１　3330



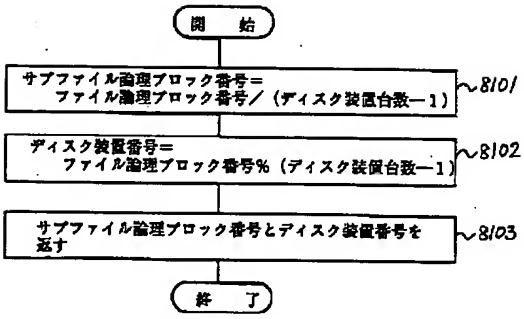
【図１２】

図１２ 3340



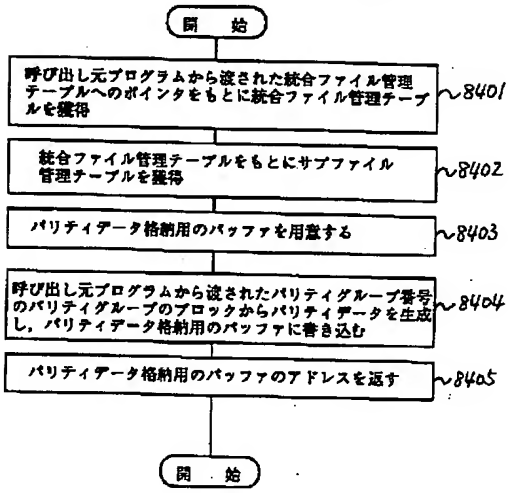
【図１３】

図１３ 3101



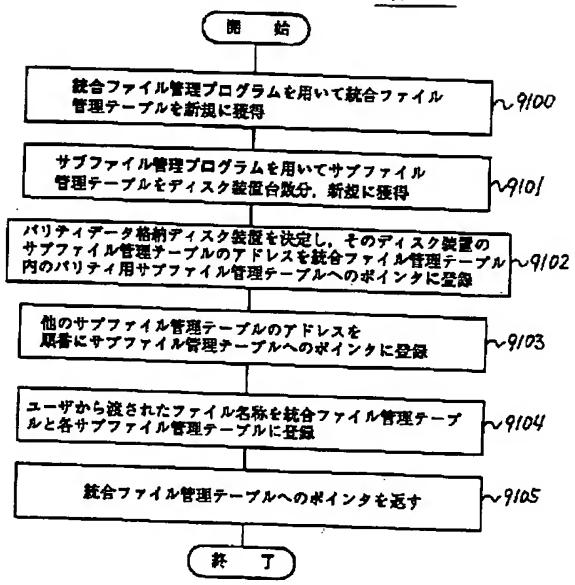
【図１４】

図１４ 3401



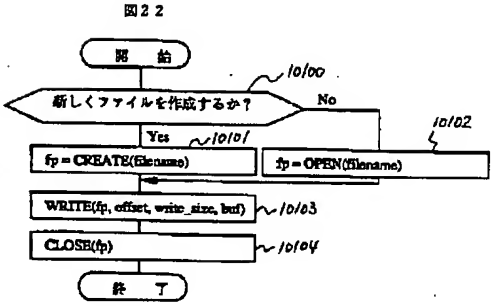
【図１６】

図１６ 7100



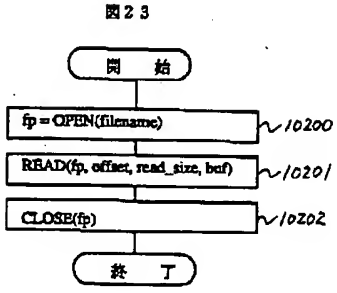
【図２２】

図２２



【図２３】

図２３



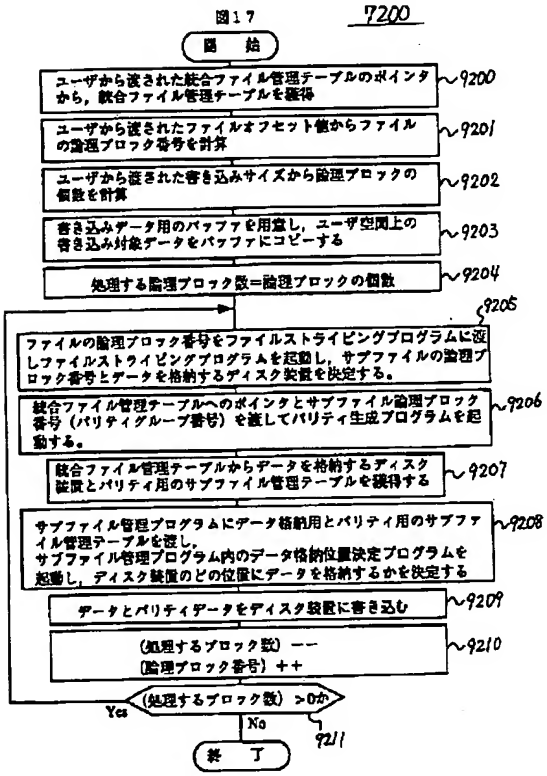
【図１５】

図１５

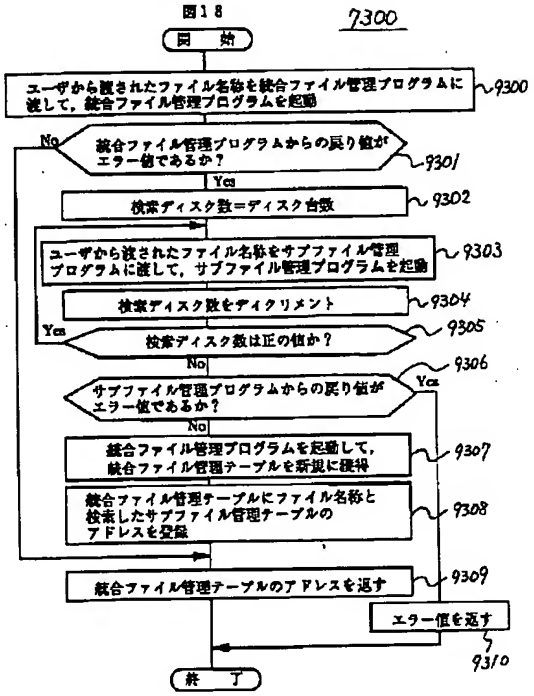
No.	コマンド名	引数	戻り値
1	CREATE( )	filename	fp
2	WRITE( )	fp, offset, write_size, buf	NULL
3	OPEN( )	filename	fp又はerror_value
4	READ( )	fp, offset, read_size, buf	NULL
5	CLOSE( )	fp	NULL
6	RECOVERY( )	fp	NULL

filename:ファイル名称,
fp:統合ファイル管理テーブルのポインタ,
offset:ファイルオフセット値,
write_size:書き込みデータサイズ,
buf:ユーザ空間上のデータバッファ,
read_size:読み出しデータサイズ,
NULL:戻り値なし

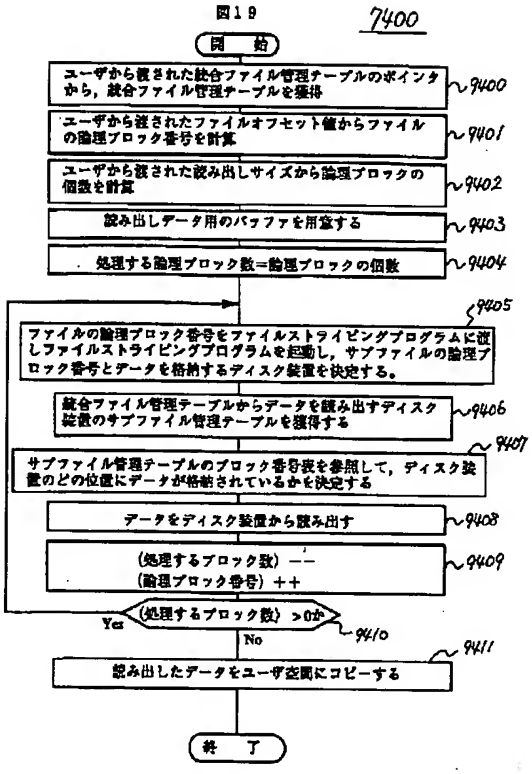
【図１７】



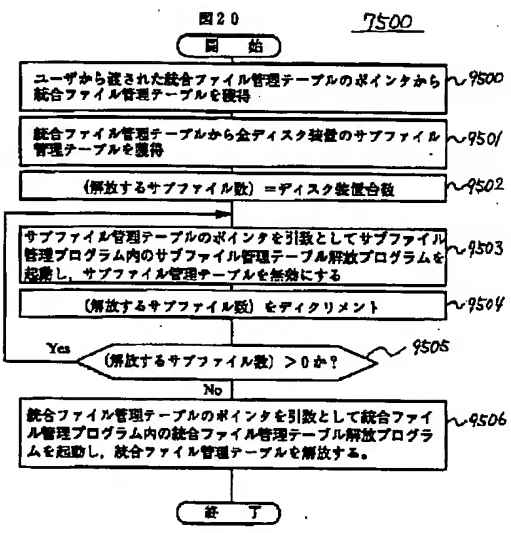
【図１８】



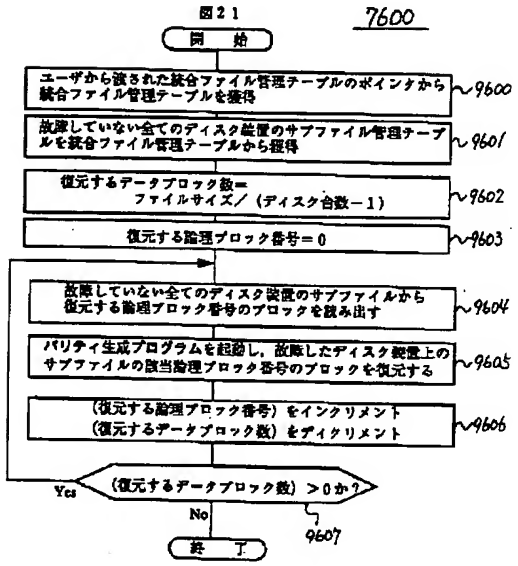
【図１９】



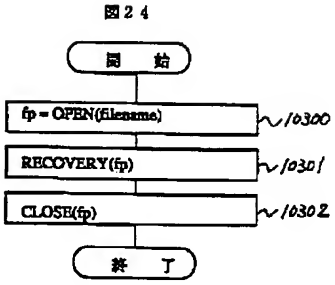
【図２０】



【図２１】

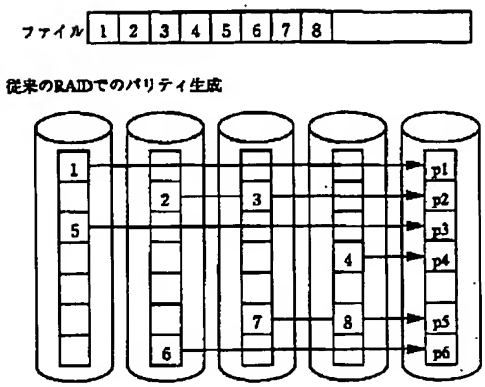


【図２４】



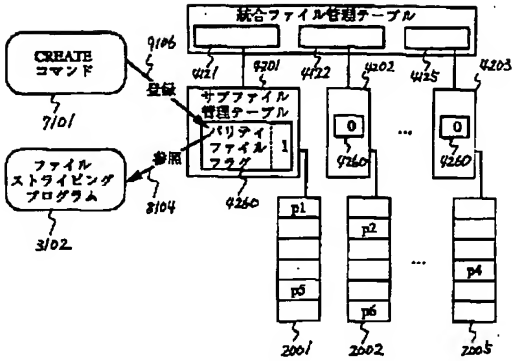
【図２５】

図２５



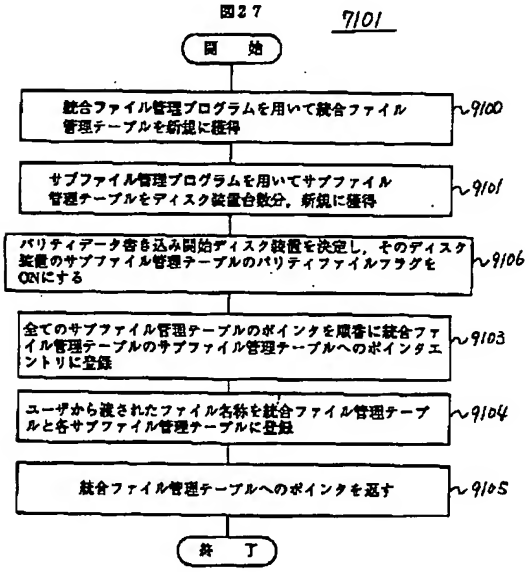
【図２６】

図２６



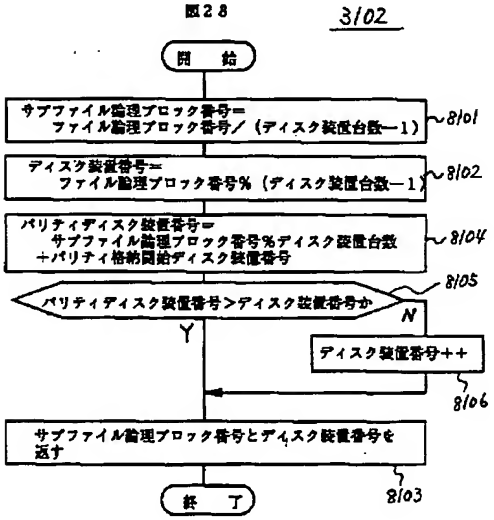
【図２７】

図２７　7101



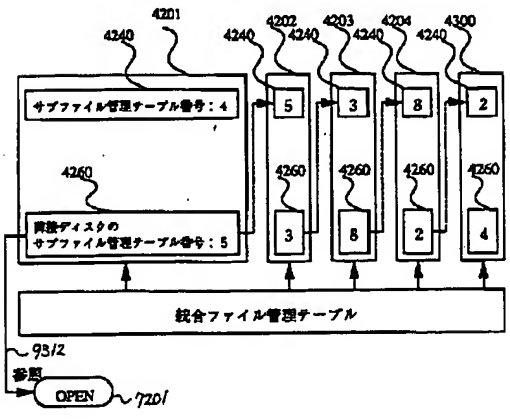
【図２８】

図２８　3102



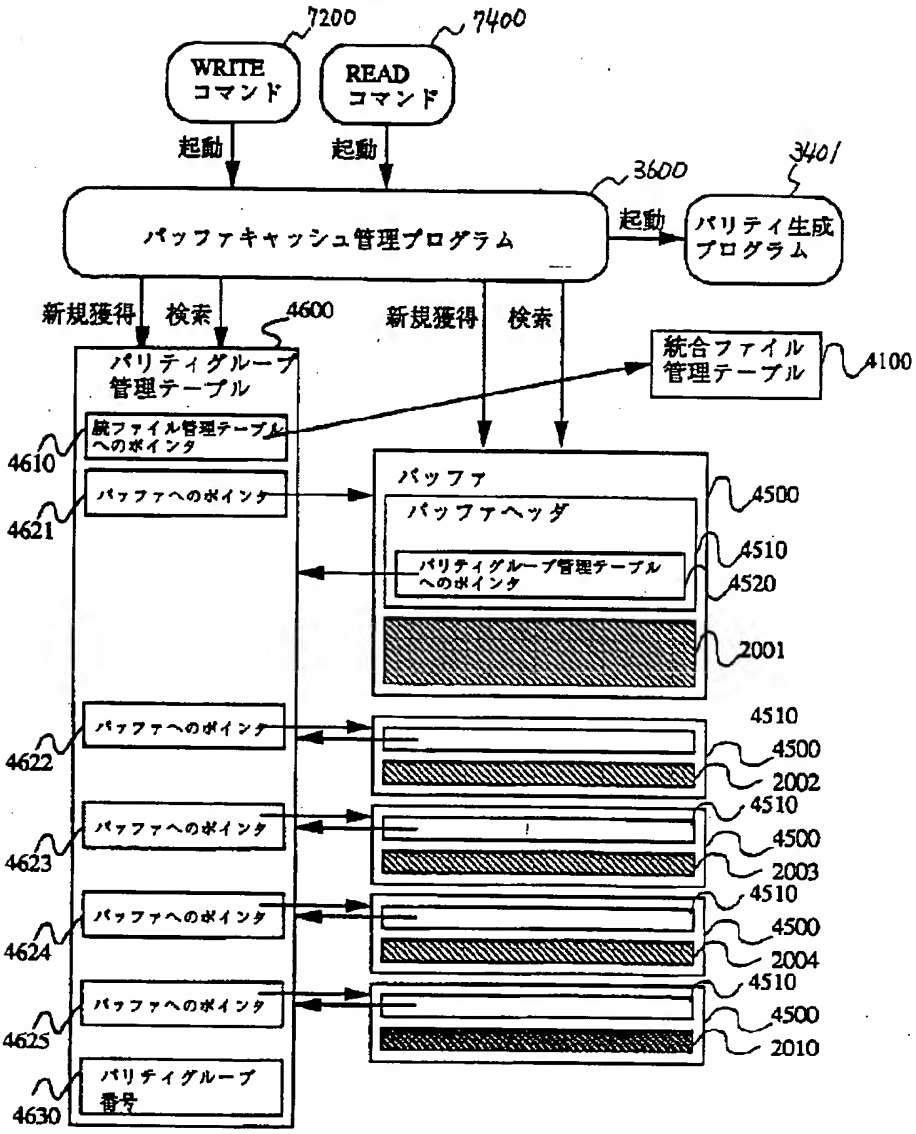
【図３３】

図３３



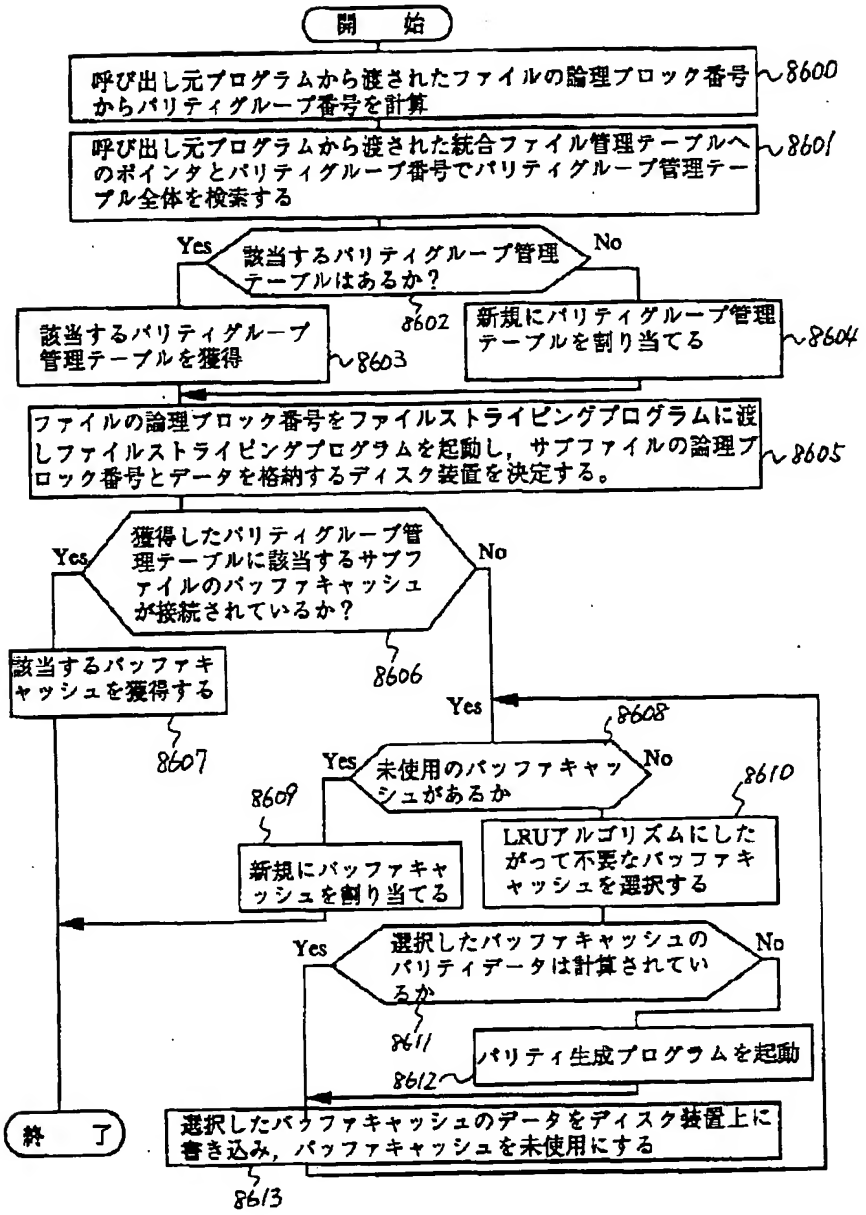
【図２９】

図２９


7200
7400
WRITE コマンド
READ コマンド
起動
起動
3600
3401
バッファキャッシュ管理プログラム
起動
パリティ生成プログラム
新規獲得
検索
4600
新規獲得
検索
パリティグループ管理テーブル
統合ファイル管理テーブル
4100
統ファイル管理テーブルへのポインタ
4610
バッファへのポインタ
4621
バッファ
バッファヘッダ
4500
パリティグループ管理テーブルへのポインタ
4510
4520
2001
バッファへのポインタ
4622
4510
4500
2002
バッファへのポインタ
4623
4510
4500
2003
バッファへのポインタ
4624
4510
4500
2004
バッファへのポインタ
4625
4510
4500
2010
パリティグループ番号
4630

【図30】

図30　3600


開　始
呼び出し元プログラムから渡されたファイルの論理ブロック番号からパリティグループ番号を計算
8600
呼び出し元プログラムから渡された統合ファイル管理テーブルへのポインタとパリティグループ番号でパリティグループ管理テーブル全体を検索する
8601
Yes
該当するパリティグループ管理テーブルはあるか？
No
8602
該当するパリティグループ管理テーブルを獲得
8603
新規にパリティグループ管理テーブルを割り当てる
8604
ファイルの論理ブロック番号をファイルストライピングプログラムに渡しファイルストライピングプログラムを起動し，サブファイルの論理ブロック番号とデータを格納するディスク装置を決定する。
8605
Yes
獲得したパリティグループ管理テーブルに該当するサブファイルのバッファキャッシュが接続されているか？
No
8606
該当するバッファキャッシュを獲得する
8607
Yes
8608
Yes
未使用のバッファキャッシュがあるか
No
8609
8610
新規にバッファキャッシュを割り当てる
LRUアルゴリズムにしたがって不要なバッファキャッシュを選択する
Yes
選択したバッファキャッシュのパリティデータは計算されているか
No
8611
パリティ生成プログラムを起動
8612
終　了
選択したバッファキャッシュのデータをディスク装置上に書き込み，バッファキャッシュを未使用にする
8613

【図３１】

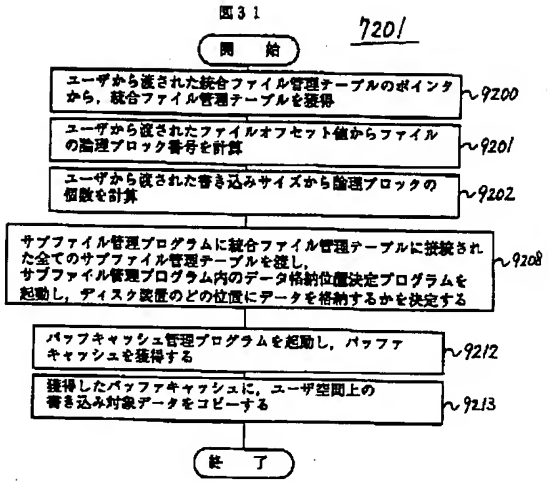
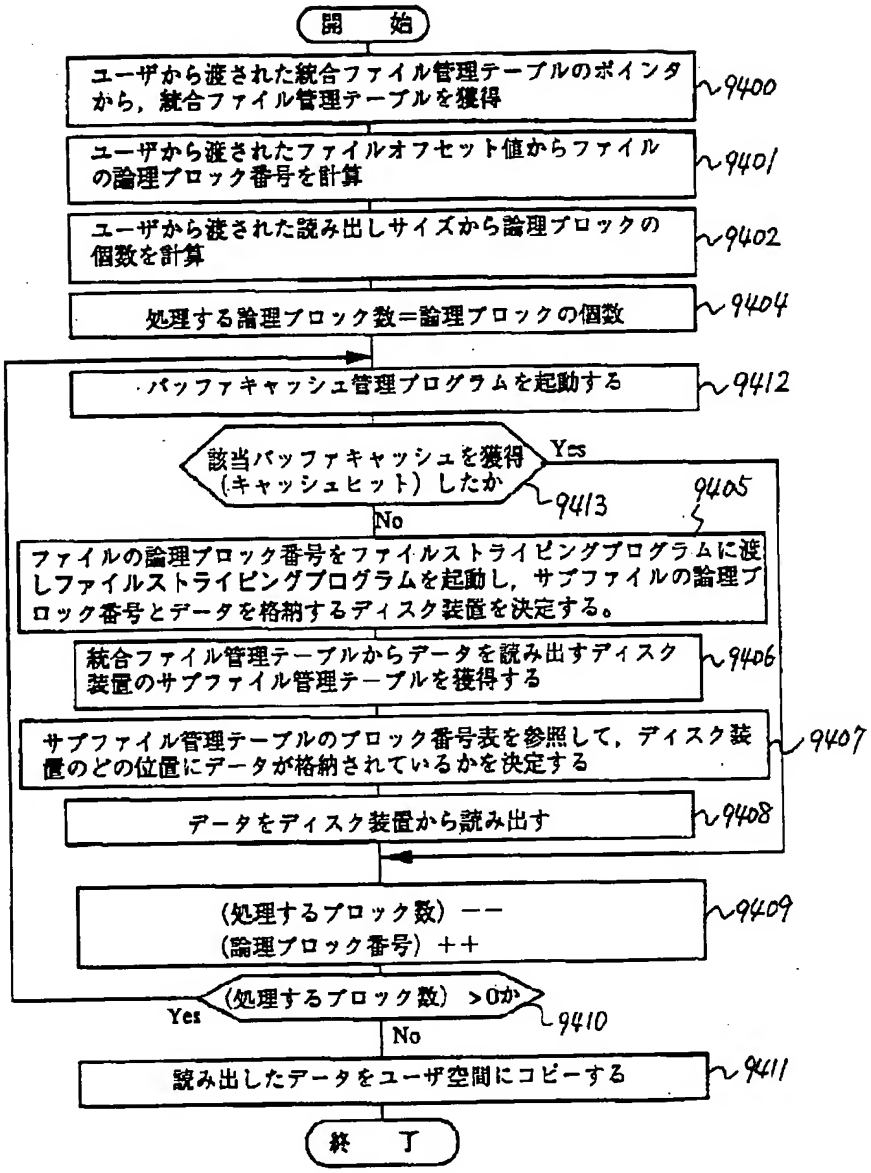


図３１
7201
開　始
ユーザから渡された統合ファイル管理テーブルのポインタから，統合ファイル管理テーブルを獲得
9200
ユーザから渡されたファイルオフセット値からファイルの論理ブロック番号を計算
9201
ユーザから渡された書き込みサイズから論理ブロックの個数を計算
9202
サブファイル管理プログラムに統合ファイル管理テーブルに接続された全てのサブファイル管理テーブルを渡し，サブファイル管理プログラム内のデータ格納位置決定プログラムを起動し，ディスク装置のどの位置にデータを格納するかを決定する
9208
バッファキャッシュ管理プログラムを起動し，バッファキャッシュを獲得する
9212
獲得したバッファキャッシュに，ユーザ空間上の書き込み対象データをコピーする
9213
終　了

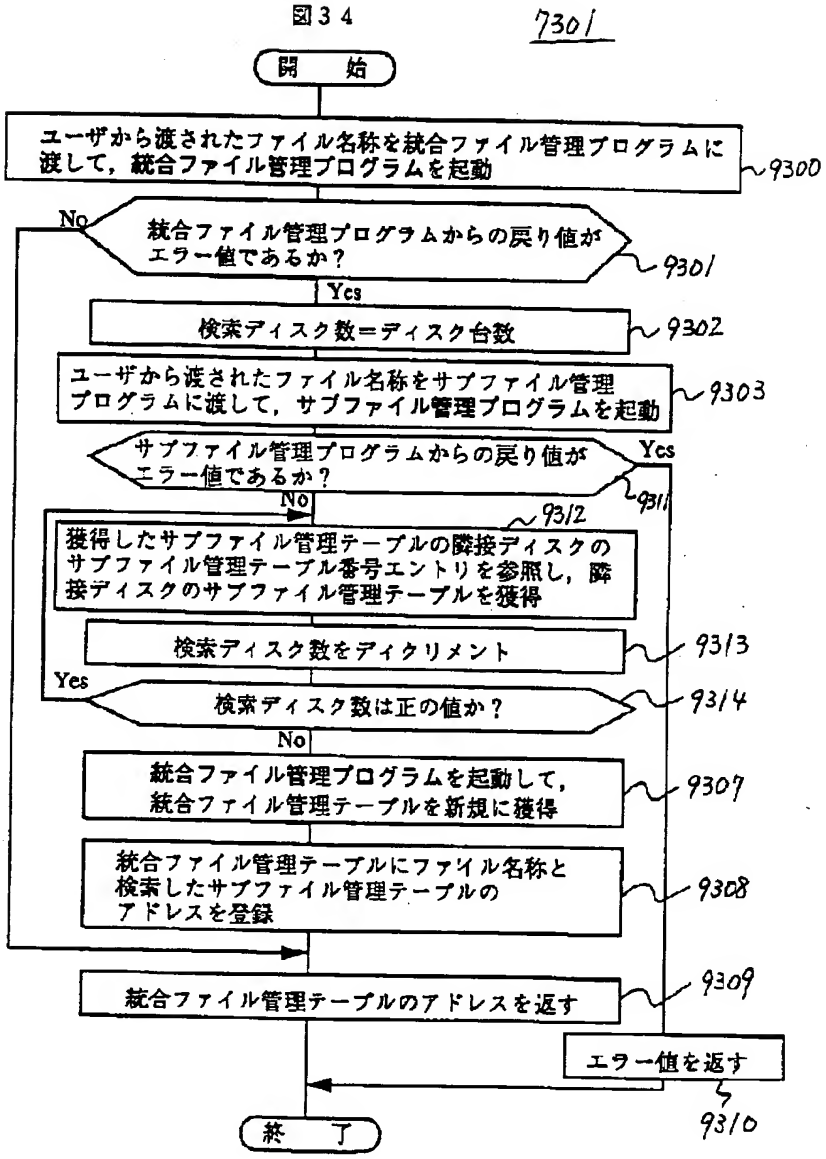
【図32】

図32　7401


開　始
ユーザから渡された統合ファイル管理テーブルのポインタから，統合ファイル管理テーブルを獲得
9400
ユーザから渡されたファイルオフセット値からファイルの論理ブロック番号を計算
9401
ユーザから渡された読み出しサイズから論理ブロックの個数を計算
9402
処理する論理ブロック数＝論理ブロックの個数
9404
バッファキャッシュ管理プログラムを起動する
9412
該当バッファキャッシュを獲得（キャッシュヒット）したか
Yes
9413
No
9405
ファイルの論理ブロック番号をファイルストライピングプログラムに渡しファイルストライピングプログラムを起動し，サブファイルの論理ブロック番号とデータを格納するディスク装置を決定する。
統合ファイル管理テーブルからデータを読み出すディスク装置のサブファイル管理テーブルを獲得する
9406
サブファイル管理テーブルのブロック番号表を参照して，ディスク装置のどの位置にデータが格納されているかを決定する
9407
データをディスク装置から読み出す
9408
（処理するブロック数）－－
（論理ブロック番号）＋＋
9409
（処理するブロック数）＞0か
Yes
9410
No
読み出したデータをユーザ空間にコピーする
9411
終　了

【図34】

図34



フロントページの続き

(72)発明者　髙橋　英男
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(72)発明者　畠山　敦
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(72)発明者　加藤　寛次
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(72)発明者　竹村　宏志
神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウエア開発本部内

(72)発明者　裏谷　郁夫
神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウエア開発本部内
(72)発明者　鬼頭　昭
神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウエア開発本部内
(72)発明者　牧　敏行
神奈川県秦野市堀山下１番地　日立コンピュータエンジニアリング株式会社内
(72)発明者　山田　秀則
神奈川県秦野市堀山下１番地　日立コンピュータエンジニアリング株式会社内
(72)発明者　城田　浩二
神奈川県秦野市堀山下１番地　日立コンピュータエンジニアリング株式会社内
(72)発明者　髙良　亜紀子
神奈川県秦野市堀山下１番地　日立コンピュータエンジニアリング株式会社内